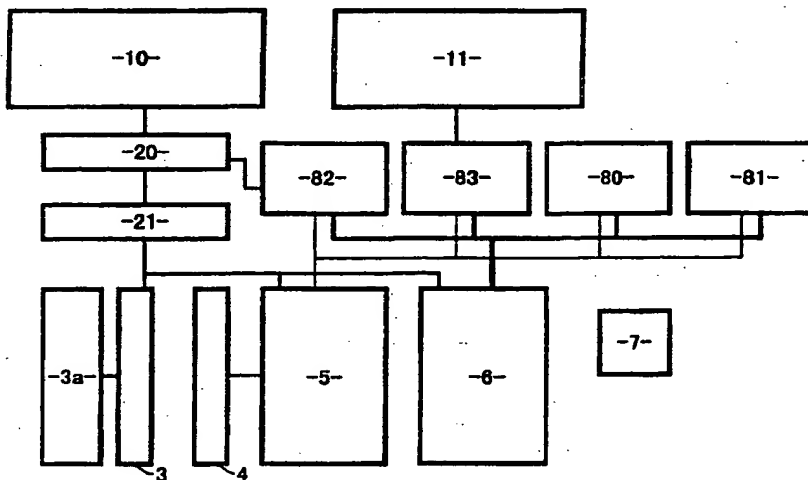




(51) 国際特許分類6 G06F 9/38, 15/82	A1	(11) 国際公開番号 WO99/27439 (43) 国際公開日 1999年6月3日(03.06.99)
(21) 国際出願番号 PCT/JP98/05230 (22) 国際出願日 1998年11月19日(19.11.98) (30) 優先権データ 特願平9/362473 1997年11月20日(20.11.97) JP 特願平10/282118 1998年10月5日(05.10.98) JP (71) 出願人 ; および (72) 発明者 関 一(SEKI, Hajime)[JP/JP] 〒790-0848 愛媛県松山市道後喜多町4番38号 Ehime, (JP)		(81) 指定国 AU, BG, BR, CA, CN, CU, CZ, HU, ID, IL, IS, JP, KR, MX, NO, NZ, PL, SG, SK, TR, UA, US, VN, ARIPO特許 (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), ユーラシア特許 (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), 欧州特許 (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI特許 (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG). 添付公開書類 国際調査報告書

(54) Title: COMPUTER SYSTEM

(54) 発明の名称 計算機システム



(57) Abstract

A computer system for processing a program described in the machine language of a stack machine, comprising a data cache (11), an integrated register file (6) in which data is written in each of entries, an advanced pointer stack (3) and a completion pointer stack (4) in which the addresses of the entries of the integrated register file are written into entries, an instruction queue (5) having the construction of an FIFO in which the contents of each of instructions are written into each entry, arithmetic units (80 and 81), and a loading/storing unit (83). When the instruction held in the first entry of the instruction queue can be completed or has been completed, the completion pointer stack is operated so as to execute the same operation of the advanced pointer stack when the held instruction is decoded on the basis of the contents of the first entry of the instruction queue, and the first entry is eliminated from the instruction queue.

(57)要約

データ・キャッシュ（１１）と各々のエントリにデータが書き込まれるようになっている統合レジスタ・ファイル（６）と、各々のエントリに統合レジスタ・ファイルのエントリのアドレスが書き込まれるようになっている前進ポインタ・スタック（３）及び完了ポインタ・スタック（４）と、各々のエントリに個々の命令の内容が書き込まれるようになっているＦＩＦＯの構成となっている命令キュー（５）と、演算ユニット（８０），（８１）と、ロード／ストア・ユニット（８３）を具備する、スタックマシンの機械語で記述されたプログラムを処理する計算機システムである。

命令キューの先頭のエントリにおいて保持されている命令の完了が可能である、あるいはそうなると、その命令キューの先頭のエントリの内容に基づき、保持されている命令がデコードされた際の前進ポインタ・スタックの動作を再現すべく完了ポインタ・スタックを操作し、命令キューからその先頭のエントリを除外するようになっている。

PCTに基づいて公開される国際出願のパンフレット第一頁に掲載されたPCT加盟国を同定するために使用されるコード(参考情報)

AE アラブ首長国連邦
AL アルバニア
AM アルメニア
AU オーストラリア
AZ アゼルバイジャン
BA ボスニア・ヘルツェゴビナ
BB バルバドス
BE ベルギー
BF ブルキナ・ファソ
BG ブルガリア
BJ ベナン
BR ブラジル
BS ベラルーシ
CA カナダ
CC 中央アフリカ
CG コンゴ
CH スイス
CI コートジボアール
CM カメルーン
CN 中国
CU キューバ
CY キプロス
CZ チェッコ
DE ドイツ
DK デンマーク
EE エストニア

ES スペイン
FI フィンランド
FR フランス
GA ガボン
GB 英国
GD グレナダ
GE ジョージア
GH ガーナ
GN ガンビア
GM ギニア
GW ギニア・ビサウ
GR ギリシャ
HR クロアチア
HU ハンガリー
ID インドネシア
IE アイルランド
IL イスラエル
IN インド
IS アイスランド
IT イタリア
JP 日本
KE ケニア
KG キルギスタン
KP 北朝鮮
KR 韓国
KZ カザフスタン
LC セントルシア

LI リヒテンシュタイン
LK スリ・ランカ
LR リベリア
LS レソト
LT リトアニア
LU ルクセンブルグ
LV ラトヴィア
MC モナコ
MD モルドヴァ
MG マダガスカル
MK マケドニア旧ユーゴスラヴィア
共和国
ML マリ
MN モンゴル
MR モーリタニア
MW マラウイ
MX メキシコ
NE ニジェール
NL オランダ
NO ノールウェー
NZ ニュー・ジージーランド
PL ポーランド
PT ポルトガル
RO ルーマニア
RU ロシア
SD スーダン
SE スウェーデン

SG シンガポール
SI スロヴェニア
SK スロヴァキア
SL シエラ・レオネ
SN セネガル
SZ スワジランド
TD チャード
TG トーゴ
TJ タジキスタン
TM トルクメニスタン
TR トルコ
TT トリニダード・トバゴ
UA ウクライナ
UG ウガンダ
US 米国
UZ ウズベキスタン
VN ヴィエトナム
YU ユーゴスラビア
ZA 南アフリカ共和国
ZW ジンバブエ

明 細 書

計算機システム

技術分野

本発明は、スタックマシンの機械語で記述されたプログラムを高速で
5 処理する新規な構成の計算機システムに関するものである。

背景技術

従来、スタックマシンにおいては、命令の実行は、基本的にプログラム上の順序通り（in-order）に行われるものであった。すなわち、スタ
10 ックマシンにおける演算命令は、オペランド・スタックからソース・データをポップし、演算を実行し、その演算結果をオペランド・スタックにプッシュするというような動作を指示するものであるが、このような命令の連鎖として書かれたプログラムを逐次的に実行するのである。

このような従来のスタックマシンにおいては、命令をプログラム上の
15 順序通り（in-order）に実行するので、制御構造が単純なもので済むという利点があるが、処理速度が制約を受けるという問題点があった。

そこで、スタックマシンの機械語で記述されたプログラムを
out-of-order で処理するような計算機方式が考案された。例えば、日本特
公平2-260082号、米国特許第5522051号や、米国特許第
20 5333320号及び米国特許第5765014号におけるプロセッサ要素がある。これらの明細書に示されるプロセッサは、処理性能の向上という点で十分ではない上に、正確な例外処理を保証する上で問題があった。

本発明は、上記問題点を解決するため創案されたものであり、正確な

例外処理を保証しつつ、スタックマシンの機械語で記述されたプログラムを out-of-order でより効率的に処理する計算機システムを提供することを目的としている。

5 発明の開示

本発明による計算機システムは、データ・キャッシュと、各々のエン
トリにデータが書き込まれるようになっている統合レジスタ・ファイル
と、各々のエントリに統合レジスタ・ファイルのエントリのアドレスが
書き込まれるようになっているスタックの構成となっている前進ポイン
10 タ・スタック及び完了ポインタ・スタックと、各々のエントリに個々の命
令の内容が書き込まれるようになっている F I F O キューの構成となっ
ている命令キューと、演算を実行するようになっている演算ユニットと、
データ・キャッシュ及び統合レジスタ・ファイルにアクセスできるよう
になっているロード／ストア・ユニットとを具備する。

15 従来のスタックマシンにおいて、スタックが, word1, word2, word3,
word4 (右端がスタックトップ) となっている状態は、本発明による計
算機システムにおいて、ポインタ・スタックが, 〈a〉, 〈b〉, 〈c〉,
〈d〉 (右端がスタックトップ) で、エントリ・アドレスが 〈a〉, 〈b〉,
〈c〉 及び 〈d〉 である統合レジスタ・ファイルの各エントリに、それ
20 ぞれ word1, word2, word3 及び word4 が保持されている状態に対応す
る。

本発明の計算機システムにおいては、命令がデコードされるごとに、
その命令の内容に応じて前進ポインタ・スタック及び統合レジスタ・ファ
イルを操作すると共にその命令の内容を命令キューに書き込むようにな
25 っている。この際、命令に含まれるオペランド・スタックに対するスタ
ック操作が、前進ポインタ・スタックに対して同様に適用される。ここ

で、1語のデータのオペランド・スタックへのプッシュ操作を、本発明の計算機システムにおいてエミュレートするには、割り付けられていない統合レジスタ・ファイルの1エントリをそのデータを保持すべく割り付け、そのエントリのアドレスを前進ポインタ・スタックにプッシュすればよい。

即ち、オペランド・スタックに対するポップ操作を含む命令がデコードされた場合には、ポップすべき語数と同じ数だけ統合レジスタ・ファイルのエントリのアドレスを前進ポインタ・スタックからポップする。オペランド・スタックに対するプッシュ操作を含む命令がデコードされた場合には、プッシュすべき語数と同じ数だけ割り付けられていない統合レジスタ・ファイルのエントリを割り付け、上記割り付けた統合レジスタ・ファイルのエントリのアドレスを前進ポインタ・スタックにプッシュする。さらに、デコードされた命令の内容を、ポップ／プッシュ操作を伴う命令の場合にはポップ／プッシュされる統合レジスタ・ファイルのエントリのアドレスと共に、命令キューに書き込むようになっている。

命令キューに保持されている未実行の命令は、データ駆動（各々の動作が、必要なすべてのソース・データが揃い実行可能となった時点で実行されること）の原理に基づき **out-of-order** で処理されるようになっている。

例えば、命令キューにおいて、演算命令を書き込みの内容とし、必要なソース・データが全て統合レジスタ・ファイルに書き込み済みとなっているエントリがあり、演算ユニットが利用できる状態であれば、その演算の実行を開始する。演算の実行が正常に終了すれば、演算結果をデスティネーションである統合レジスタ・ファイルのエントリに書き込む。

命令キューの先頭のエントリにおいて保持されている命令の完了が可能である、あるいはそうなると、その命令キューの先頭のエントリの内

容に基づき、保持されている命令がデコードされた際の前進ポインタ・スタックの動作を再現すべく完了ポインタ・スタックを操作し、命令キューからその先頭のエントリを除外し、ポップ操作によって完了ポインタ・スタックにおけるアドレスの保持が無くなった統合レジスタ・ファイルのエントリの割り付けを解除するようになっている。

図面の簡単な説明

第1図は、本発明にかかる好ましい計算機システムの基本構成を示すブロック図、第2図は、前進ポインタ・スタック及び完了ポインタ・スタックの構成を示す図、第3図は、後述する本発明第1実施例における統合レジスタ・ファイルの各々のエントリの詳細な構成を示す図、第4図は、命令キューの構成を示す図、第5図は、命令キューの各々のエントリの詳細な構成を示す図、第6図～第12図は、本発明第1実施例における一動作例の、サイクル毎の前進ポインタ・スタック、完了ポインタ・スタック、命令キュー及び統合レジスタ・ファイルの内容を具体的に示した説明図、第13図は、本発明第1実施例において、1サイクル当り3命令までデコードできるような構成をとる場合に、プログラムがどのように変換されるかを具体的に示す図表、第14図は、後述する本発明第2実施例における統合レジスタ・ファイルの各々のエントリの詳細な構成を示す図、第15図～第21図は、本発明第2実施例における一動作例の、サイクル毎の前進ポインタ・スタック、完了ポインタ・スタック、命令キュー及び統合レジスタ・ファイルの内容を具体的に示した説明図である。

25 発明を実施するための最良の形態

以下に、本発明にかかる好ましい計算機システムについて、図面を参

照しながら説明する。なお、以下に述べる本発明による計算機システムの実施例は、Java Virtual Machine (Java VM) で規定されるスタックマシンの基本的な命令をハードウェアで実行するものである。すなわち、データ語長を 32 ビットとして、これを単位にロード/ストア及び算術論理演算等の演算を行う。従って、例えば、倍長語の間での算術演算は、2 語ずつ合せて 4 語のソース・データをもとに 2 語の演算結果を生ずる。

従来のスタックマシンにおける、語の単位でデータがプッシュ/ポップされるようになっているスタックは、後述するポインタ・スタックと区別するために、以降では、ワード・スタックと呼ぶことにする。

10 Java VM にはもともとハードウェアで実行することを想定していない複雑な命令が含まれるが、以下に述べる本発明による計算機システムの実施例は、次のような基本的な命令をハードウェアで実行するものとする。

(a) 即値データのオペランド・スタックへのプッシュ命令

15 bipush, sipush, aconst_null, iconst_m1, iconst_<i>, fconst_<f>, lconst_<l>, dconst_<d>

(b) 変数データのオペランド・スタックへのロード命令

ldc1, ldc2, iload, iload_<n>, fload, fload_<n>, aload, aload_<n>, ldc2w, lload, lload_<n>, dload, dload_<n>, iaload, laload, faload, daload, aaload, baload, caload, saload

(c) オペランド・スタック上のデータの変数へのストア命令

istore, istore_<n>, fstore, fstore_<n>, astore, astore_<n>, lstore, lstore_<n>, dstore, dstore_<n>, iastore, lastore, fastore, dastore, aastore, bastore, castore, sastore

25 (d) 演算命令

(d-1) 算術演算命令

iadd, ladd, fadd, dadd, isub, lsub, fsub, dsub, imul, lmul, fmul, dmul, idiv, ldiv, fdiv, ddiv, irem, lrem, frem, drem, ineg, lneg, fneg, dneg

(d-2) 論理演算命令

ishl, ishr, iushr, lshl, lshr, lushr, iand, land, ior, lor, ixor, lxor

5 (d-3) 変換演算命令

i2l, i2f, i2d, l2i, l2f, l2d, f2i, f2l, f2d, d2i, d2l, d2f, int2byte, int2char, int2short

(d-4) 比較演算命令

lcmp, fcmpl, fcmpg, dcmpl, dcmpg

(e) オペランド・スタックの操作命令

10 pop, pop2, dup, dup2, dup_x1, dup2_x1, dup_x2, dup2_x2, swap

(f) 分岐命令

ifeq, ifnull, iflt, ifle, ifne, ifnonnull, ifgt, ifge, if_icmpeq, if_icmpne, if_icmplt, if_icmpgt, if_icmple, if_icmpge, goto, goto_w

以降、特にことわらない限り、「命令」とは上に挙げた命令のいずれ

15 かを意味するものとする。

以下に、オペランド・スタックの操作命令の処理方法の異なる、第1及び第2の2つの実施例について説明する。

まず、本発明第1実施例の計算機システムについて説明する。

20 第1図は計算機システムのブロック図であって、10は命令キャッシュ、11はデータ・キャッシュ、20は命令フェッチ・ユニット、21は命令デコード・設定ユニット、3は前進ポインタ・スタック、3aは前進ポインタ・スタック履歴ファイル、4は完了ポインタ・スタック、5は命令キュー、6は統合レジスタ・ファイル、7はフリー・リスト、80及び81は各々演算ユニット0及び1、82は分岐ユニット、83はロード

25 /ストア・ユニットを表している。

次に、本発明第1実施例の計算機システムの各構成要素ごとにその詳

細な構成を説明する。

(A) 命令フェッチ・ユニット

命令フェッチ・ユニットは、図示していないプログラムカウンタ (pc レジスタ) を具備しており、命令キャッシュから命令をフェッチし、命令
5 デコード・設定ユニットに渡す。分岐の予測や分岐の実行も担う。

(B) 命令デコード・設定ユニット

命令デコード・設定ユニットは、命令フェッチ・ユニットから渡された命令のデコードを行い、命令に含まれる演算等がデータ駆動で実行されるように、後述する前進ポインタ・スタック、命令キュー及び統合レジ
10 スタ・ファイル等を設定するための各種信号を発生する。

(C) ポインタ・スタック

ポインタ・スタックは、各々のエントリに統合レジスタ・ファイルのエントリのアドレスが書き込まれるようになっているスタックの構成となっている。

15 従来のスタックマシンにおいて、ワード・スタックが, word1, word2, word3, word4 (右端がスタックトップ) となっている状態は、本発明による計算機システムにおいて、ポインタ・スタックが, 〈a〉, 〈b〉, 〈c〉, 〈d〉 (右端がスタックトップ) で、エントリ・アドレスが 〈a〉, 〈b〉, 〈c〉 及び 〈d〉 である統合レジスタ・ファイルの各エ
20 ントリに、それぞれ word1, word2, word3 及び word4 が保持されている状態に対応する。

本発明の計算機システムは、前進ポインタ・スタック (A P S ; Advanced Pointer Stack) と完了ポインタ・スタック (C P S ; Completed Pointer Stack) の2つのポインタ・スタックを具備する。

25 本発明の計算機システムにおいては、個々の命令がデコードされるごとに、前進ポインタ・スタック (以下ではA P Sで示す) 及び統合レジ

スタ・ファイル进行操作すると共に命令の内容を命令キューに書き込むことにより、命令に含まれる演算等がデータ駆動で実行されるべく設定される。すなわち、前進ポインタ・スタックはデコード・設定済みの全ての命令によるスタック操作を反映している。

- 5 他方、完了ポインタ・スタック（以下ではC P Sで示す）は、プログラム上の順番で完了済みの全ての命令によるスタック操作を反映するものである。本発明の計算機システムはデータ駆動の原理に基づく out-of-order 実行を可能とするものであるが、完了ポインタ・スタックは、正確な例外処理を保証するため、プログラムが in-order で実行された場合
- 10 の状態を構成するために存在するものである。

- ポインタ・スタック及び統合レジスタ・ファイルのエントリ数は限られたものであるので、ワード・スタックが成長すると、ポインタ・スタック及び統合レジスタ・ファイルを用いてスタック・トップ近傍の部分しか保持できない。本実施例の計算機システムにおいては、ワード・スタック
- 15 の残りの部分はデータ・キャッシュに格納されるようになっている。そのため、各ポインタ・スタックは循環型のバッファの構成となっており、プッシュ・ポインタとボトム・ポインタと呼ぶ2つのレジスタが各々存在する。プッシュ・ポインタは、統合レジスタ・ファイルのエントリのアドレスを保持する最上位のエントリの1つ上を示す。ボトム・ポインタは、
- 20 統合レジスタ・ファイルのエントリのアドレスを保持する最下位のエントリを示す。ボトム・ポインタの値からプッシュ・ポインタの値を引くことで、ポインタ・スタックに何エントリの空きがあるかがわかる。初期状態においては、プッシュ・ポインタ及びボトム・ポインタの各々の値は共に0となっている。

- 25 第2図は、本実施例の計算機システムにおける各ポインタ・スタックと各プッシュ・ポインタ及びボトム・ポインタの関係を示す説明図であ

る。2つのポインタ・スタックAPS 3及びCPS 4は同数のエントリを有し、各ポインタ・スタックで各々のエントリに下から順に0、1、2、...とアドレスが付けられているものとする。縦線が施されているエントリは統合レジスタ・ファイルのエントリのアドレスを保持しているものとする。第2図に示すように、プッシュ・ポインタは、APS及びCPSの各々に対して設けられており、それぞれPP_OF_APS及びPP_OF_CPSと名付けている。他方、ボトム・ポインタは1つだけ設けられており、これがAPS及びCPSで共用される。これをBP_OF_PSと名付けている。

APSとCPSの間には、エントリの数だけ比較回路が設けられており、APS及びCPSの同じエントリ・アドレスにある（第2図において水平に並ぶ）エントリの間でその内容が比較されるようになっている。

命令に含まれるオペランド・スタックに対する1語分のプッシュ操作に対応して割り付けられる統合レジスタ・ファイルの1エントリのアドレスをAPSのPP_OF_APSで示されるエントリに書き込み、PP_OF_APSの値に1を加えるようになっている。逆に、命令に含まれるオペランド・スタックに対する1語分のポップ操作に対応して、PP_OF_APSの値から1を引くようになっている。CPSとPP_OF_CPSに関しても同様である。

BP_OF_PSで示されるエントリの内容がAPSとCPSで一致する場合には、その2つのポインタ・スタックで一致する内容で示される統合レジスタ・ファイルのエントリに書き込まれている1語分のデータをデータ・キャッシュにストア（Spill）することができる。その際、BP_OF_PSの値に1を加えるようになっている。逆に、データ・キャッシュにストア（Spill）したデータを統合レジスタ・ファイルにロード（Fill）する場合には、最後にストア（Spill）した1語分のデータに対し、フリー・リ

ストに登録されている統合レジスタ・ファイルの1エントリを割り付けてそのデータを書き込み、その統合レジスタ・ファイルのエントリのアドレスをAPS及びCPSのBP_OF_PSで示されるエントリの1つ下に各々書き込み、BP_OF_PSの値から1を引くようになっている。

- 5 本実施例の計算機システムは、分岐予測に基づく投機的実行を可能にするために、前進ポインタ・スタック履歴ファイル（以下では「APS履歴ファイル」と記す）を具備する。APS履歴ファイルの各々のエントリには、APSの全エントリ及びPP_OF_APSの内容が書き込めるようになっている。

10 (D) 統合レジスタ・ファイル (CRF ; Consolidated Resister File)

統合レジスタ・ファイル（以下ではCRFで示す）は、従来のスタックマシンにおけるワード・スタックの内容を、順序不同で保持するものである。

- 15 第3図は、本第1実施例における、CRF 6の各々のエントリ 6 (i)の詳細な構成を示す説明図である。ここで、iはエントリのアドレスである。CRF 6の各々のエントリ 6 (i)はデータ・フィールド 6 1 (i)、書き込み完了フラグ (WCF, Write Completion Flag) フィールド 6 2 (i)、カラー (C, Colour) フィールド 6 3 (i)及びビジービット (BB) フィールド 6 4 (i)から成っている。

- 20 実際のCRFのハードウェア上の構成は、上述の各フィールド別に設けられたレジスタ・ファイルの集合体である。

CRFの各々のエントリのデータ・フィールドは、1語分のデータが書き込まれる構成となっている。

- 25 CRFの各々のエントリにおいて、WCFフィールドは、データ・フィールドにデータの書き込みが完了していれば1、完了していなければ0が書き込まれているようになっている。

CRFの各々のエントリにおいて、Cフィールドは、そのCRFのエントリが、命令に含まれるプッシュ操作に対応して割り付けられたものであるのか、アンダーフロー回避のためのデータ・キャッシュからのロード(Fill)の際に割り付けられたものであるのかの区別、前者の場合にはさらに分岐タグが書き込まれるようになっている。本実施例においては、後述するように、分岐タグはAPS履歴ファイルのエントリのアドレスと一定の関係にある。

CRFの各々のエントリにおいて、BBフィールドは、そのCRFのエントリがデータを保持すべく割り付けられている状態であれば1、割り付けられていない状態であれば0が書き込まれているようになっている。

(E) フリー・リスト (FL)

フリー・リスト (以下ではFLで示す) は、フリーな、即ち、割り付けられていない (BBフィールドが0である) CRFのエントリのアドレスを保持するためのメモリであり、本実施例においては、循環型のFIFOキューの構成となっている。

初期状態においては、CRFの全てのエントリのアドレスがFLに登録されている。CRFのフリーなエントリを割り付ける必要がある場合に、FLからフリーなCRFのエントリのアドレスが取り出される。逆に、CRFのあるエントリの割り付けが解除されれば、そのエントリのアドレスがFLに登録されるようになっている。

(F) 命令キュー (IQ ; Instruction Queue)

命令キュー (以下ではIQで示す) は、実行または完了を待っているデコード・設定済の命令を保持するメモリであり、循環型のFIFOキューの構成となっている。

第4図は、IQの構成を示す説明図である。第4図において、IQ5

の各々のエントリは下から順に 0、1、2、・・・とアドレスが付けられているものとし、縦線が施されている I Q 5 のエントリは、実行または完了を待っている命令を保持しているものとする。I Q は、設定ポインタ／完了ポインタと名付けた二つのレジスタを具備する。設定ポインタは、次にデコード・設定される命令の内容を書き込むべきエントリを示す。完了ポインタは、次に完了されるべき命令のエントリを示す。完了ポインタの値から設定ポインタの値を引くことで、I Q に何エントリの空きがあるかがわかる。初期状態においては、設定ポインタ及び完了ポインタの値は共に 0 となっている。

- 10 第 5 図は、I Q 5 の各々のエントリ 5 (i) の詳細な構成を示す説明図である。ここで、i はエントリのアドレスである。I Q 5 の各々のエントリ 5 (i) はオペレーション・フィールド 5 0 0 (i)、オペランド・フィールド 5 0 1 (i)、第 1 ソース・フィールド 5 1 0 (i)、第 1 書込み完了フラグ (WCF 1) フィールド 5 1 1 (i)、第 2 ソース・フィールド 5 2 0 (i)、
15 第 2 書込み完了フラグ (WCF 2) フィールド 5 2 1 (i)、第 3 ソース・フィールド 5 3 0 (i)、第 3 書込み完了フラグ (WCF 3) フィールド 5 3 1 (i)、第 4 ソース・フィールド 5 4 0 (i)、第 4 書込み完了フラグ (WCF 4) フィールド 5 4 1 (i)、第 1 デスティネーション・フィールド 5 5 (i)、第 2 デスティネーション・フィールド 5 6 (i)、分岐タグ (BT) フィールド 5 7 (i)、及び実行状態 (S ; State) フィールド 5 8 (i) から
20 成っている。

I Q の各々のエントリのオペレーション・フィールドはオペレーション・コードが書き込まれる構成となっている。

- 25 I Q の各々のエントリのオペランド・フィールドは、オペレーション・コードに続いてオペランドが示されるような命令の場合に、このオペランドが書き込まれるようになっている。

I Qの各々のエントリの第1～第4ソース・フィールドの各々は、ソース・データを保持すべく割り付けられているCRFのエントリのアドレスが書き込まれるようになっている。オペランド・スタックに対するポップ操作を含む命令の場合には、命令によりポップされるべきデータを保持すべく割り付けられているCRFのエントリのアドレスが、ポップされる順に書き込まれるようになっている。

I Qの各々のエントリの第1～第2デスティネーション・フィールドの各々は、命令のデコード・設定に伴い、新たに割り付けられるCRFのエントリのアドレスが書き込まれるようになっている。オペランド・スタックに対するプッシュ操作を含む命令の場合には、命令によりプッシュされるべきデータを保持すべく割り付けられるCRFのエントリのアドレスが、プッシュされる順に書き込まれるようになっている。

I Qの各々のエントリにおいて、第1～第4の各WCFフィールドは各々第1～第4ソース・フィールドに対応して設けられている。WCF 1フィールドは第1ソース・フィールドに示されるCRFのエントリにデータの書き込みが完了していれば1、完了していなければ0が書き込まれているようになっている。第2～第4のWCFフィールド、ソース・フィールドに関しても同様である。

I Qは、各エントリのソース・フィールドごとに比較回路を備えており、データの書き込みが行われるCRFのエントリのアドレスを各ソース・フィールドの内容と比較して、一致するソース・フィールドに対応するWCFフィールドに1を立てるような機能を有する。

I Qの各々のエントリのBTフィールドは、分岐予測に基づく投機的実行に係るもので、本実施例においては、後述するように、BTフィールドに書き込まれる分岐タグはAPS履歴ファイルのエントリのアドレスと一定の関係にある。

I Qの各々のエントリにおいて、Sフィールドは、そのエントリに書き込まれている命令の実行状態に応じて、未実行、実行済み、正常終了、例外事象発生等の情報が書き込まれているようになっている。

(G) 演算ユニット

- 5 本実施例の計算機システムは、演算ユニット0及び演算ユニット1を具備しており、その各々は、I Qより送られてくる算術論理演算、データ・タイプの変換演算、比較演算等の演算命令を実行する機能を有し、互いに独立に並行して動作することができるようになっている。

- 10 本発明の計算機システムにおいては、各々の演算ユニットをパイプライン化したり、より多くの演算ユニットを具備したり、演算ユニットごとに実行する演算の種類を特定した構成とすることも可能である。

(H) 分岐ユニット

分岐ユニットは、I Qより送られてくる条件分岐命令を処理し、分岐の有無を確定して、命令フェッチ・ユニットに通知する機能を有する。

- 15 (I) ロード/ストア・ユニット (LSU ; Load/Store Unit)

ロード/ストア・ユニット (以下ではLSUで示す) は、アドレス計算を行う機能を有し、データ・キャッシュ及びCRFにアクセスすることができるようになっている。

- 20 LSUは、最初のローカル変数へのポインタを保持する図示していないレジスタ (vars レジスタ) を具備する。本実施例の計算機システムにおいては、最初のローカル変数の格納域はデータ・キャッシュあるいはCRFにあるが、vars レジスタには、データ・キャッシュにおける相当するアドレス値が書き込まれているようになっている。すなわち、全てあるいは一部のローカル変数の実際の格納域がCRFにある場合でも、
25 各々のローカル変数に、全てのローカル変数をデータ・キャッシュにストア (Spill) したと仮定した場合のデータ・キャッシュにおけるアドレ

5 ス値を対応させることができるので、ロード／ストア命令の処理において、LSUは vars レジスタの値を用いてアドレス計算を行い、対象となるローカル変数の格納域がデータ・キャッシュかCRFかを判定し、その格納域にアクセスする。格納域がCRFにあると判定された場合には、アクセスすべきCRFのエントリのアドレスはAPSから読み出される。

ロード／ストア命令が命令デコード・設定ユニットでデコードされると、その内容が、命令キューに書き込まれると同時に、LSUにも送られるようになっている。

10 LSUは、プログラム上の順番で、ロード／ストア命令を命令キューのエントリのアドレス、対象となる変数名及びデータと共に蓄える図示してないキューを具備する。このキューは連想機能を備えており、変数名を照合してデータ・アクセスの依存性の検証を行うことにより、ロード命令の実行を out-of-order で行うことができるようになっている。本
15 実施例の計算機システムにおいては、ローカル変数の格納域はデータ・キャッシュあるいはCRFにあるが、上記キューにすでに同じ変数名の書き込みがあるローカル変数のロード命令の場合、変数データは上記キューより読み出される。ストア命令の実行は、正確な例外処理を保証するために、後述するように、完了ステージにおいて in-order で行われる。

20 LSUは、プログラム中に示されるロード／ストア命令を実行すると共に、オーバーフロー／アンダーフローの回避のため、APS、CPS及びCRFの空きに応じて、CRFに保持されているスタックの最下位にあたるデータをデータ・キャッシュとの間で自動的にストア(Spill)／ロード(Fill)するようになっている。

25 本発明第1実施例の計算機システムにおいては、APS、CPS及びCRFのオーバーフロー／アンダーフローは以下に示すようなメカニ

ズムにより回避される。

A P S、C P SあるいはC R Fの空きが一定量以下になると、オーバーフローを回避するために以下のような制御動作が随時行われるようになっている。

- 5 すなわち、A P SがC R Fのエントリのアドレスを一定量以上保持していない場合、あるいは、B P _O F _P S で示されるエントリの内容がA P SとC P Sで一致しない場合、以上の条件が解消するまでの間、命令シーケンスの流れを停止するよう命令フェッチ・ユニットに信号が送られる。

- 10 逆に、A P SがC R Fのエントリのアドレスを一定量以上保持しており、B P _O F _P S で示されるエントリの内容がA P SとC P Sで一致する場合には、その2つのポインタ・スタックで一致する B P _O F _P S で示されるエントリの内容で示されるC R Fのエントリに書き込まれている1語分のデータをデータ・キャッシュにストア (Spill) し、B P _O F _P S の
15 値に1を加える。さらに、上記C R FのエントリのB Bフィールドを0に変更し、そのエントリのアドレスをF Lに登録する。

- 20 A P S、C P S及びC R Fの空きがいずれも一定量以上になると、アンダーフローを回避するために、最後にストア (Spill) した1語分のデータをデータ・キャッシュから取り出し、それにF Lに登録されているフリーなC R Fの1エントリを割り付け、そのデータ・フィールドに書き込む。W C F、B Bの各フィールドは1とする。さらに、その割り付けられたC R Fのエントリのアドレスを、A P S及びC P SのB P _O F _P S で示されるエントリの1つ下に各々書き込み、B P _O F _P S の値から1を引く。

- 25 また、C R Fとデータ・キャッシュの間の Spill/Fill の動作を効率的に行うために、L S Uがデータをいったん蓄えるバッファを備え、この中

に適当な語数のデータを溜めておくような構成とすることも可能である。

ついで、本発明第1実施例の計算機システムの動作を説明する。

5 本実施例の計算機システムは命令を、①命令フェッチ、②命令デコード・設定、③実行、④完了の4ステージで処理する。当分の間、説明を簡単にするため、1サイクルで1つの命令をデコード・設定／完了できるものとして、以下に各ステージごとに動作内容を説明する。

① 命令フェッチ・ステージ

10 このステージでは、命令フェッチ・ユニットが命令キャッシュから命令を取り出すと共に、次にフェッチする命令のアドレスを決定する。次に命令をフェッチするのは通常次アドレス値からであるが、フェッチした命令が無条件分岐命令であるか、条件分岐命令で分岐すると予測した場合、分岐予測が外れた場合、あるいは例外が発生した場合には、フェッチするアドレス値を変更する。

15 ② 命令デコード・設定ステージ

このステージでは、命令をデコードして、命令の内容に応じて前進ポインタ・スタック（APS）及び統合レジスタ・ファイル（CRF）を操作すると共に命令の内容を命令キュー（IQ）に書き込むことにより、命令に含まれる演算等がデータ駆動で実行されるべく設定する。以下に、
20 設定動作を詳細に説明する。

本発明の計算機システムにおいては、従来のスタック・マシンにおけるワード・スタックのスタックトップ近傍がポインタ・スタックとCRFによって再現されるが、命令に含まれるオペランド・スタックに対するスタック操作が、APSに対して同様に適用される。

25 1語のデータのオペランド・スタックへのプッシュ操作をエミュレートするには、FLに登録されているフリーなCRFの1エントリをその

データを保持すべく割り付け、そのエントリのアドレスをAPSにプッシュすればよい。

- オペランド・スタックの操作命令 (Java VM における pop, pop2, dup, dup2, dup_x1, dup2_x1, dup_x2, dup2_x2, swap) の場合、基本的には、オペランド・スタックに対して行うべき操作をAPSに対して同様に行えばよい。本第1実施例においては、スタック上でコピーを作成するようなオペランド・スタックの操作命令 (Java VM における dup, dup2, dup_x1, dup2_x1, dup_x2, dup2_x2) の場合には、コピー・データを保持すべくFLに登録されているフリーなCRFのエントリを割り付け、そのエントリのアドレスをAPSの適切なエントリに書き込むようになっている。

- 命令のデコード・設定に伴い新たに割り付けられるCRFのエントリにおいては、BBフィールドに1を立て、Cフィールドには命令デコード・設定ユニットから送られてくる分岐タグを書き込む。即値データのプッシュ命令の場合には、データがすでに得られているので、データ・フィールドにそのデータを書き込み、WCFフィールドに1を立てる。それ以外の場合には、データはデコード・設定の時点では得られていないので、WCFフィールドを0としておく。

- デコードされた命令の内容をプログラム上の順番でIQに保持しておくために、その命令の内容をIQの設定ポインタで示されるエントリに書き込み、設定ポインタの値に1を加える。すなわち、オペレーション・フィールドにオペレーション・コードを書き込み、オペレーション・コードに続いてオペランドが示されるような命令の場合には、オペランド・フィールドにこのオペランドを書き込む。BTフィールドには命令デコード・設定ユニットから送られてくる分岐タグを書き込む。Sフィールドに関しては、無条件分岐命令、即値データのオペランド・スタック

へのプッシュ命令あるいはスタック上でコピーを作成することのないオペランド・スタックの操作命令（Java VM における pop, pop2, swap）の場合は実行済みとし、その他の命令の場合は未実行としておく。

5 オペランド・スタックに対するポップ操作を含む命令の場合には、ポップすべき語数と同じ数だけ APS からポップされる CRF のエントリ
のアドレスを、その順で第 1 ～ 第 4 ソース・フィールドに書き込む。この際、エントリ・アドレスがポップされる CRF のエントリの各々で WCF フィールドを読み出し、IQ の対応する WCF フィールドに送る。

10 オペランド・スタックに対するプッシュ操作を含む命令の場合には、プッシュすべき語数と同じ数だけ APS にプッシュされる CRF のエントリ
のアドレスを、その順で第 1 ～ 第 2 デスティネーション・フィールドに書き込む。

15 本第 1 実施例においては、スタック上でコピーを作成するようなオペランド・スタックの操作命令の場合には、コピー元となるデータを保持
すべく割り付けられている CRF のエントリのアドレスをソース・フィールドに、コピー・データを保持すべく新たに割り付けられる CRF の
エントリのアドレスをデスティネーション・フィールドに、一定の対応関係のもとに書き込む。

20 命令の種類に応じて、オペランド・スタックに対してポップ／プッシュすべき語数（オペランド・スタックの操作命令の場合には、作成する
コピーの語数）は決まっているので、オペレーション・フィールドの内容によって、第 1 ～ 第 4 ソース・フィールド、第 1 ～ 第 4 の WCF フィ
ールド及び第 1 ～ 第 2 デスティネーション・フィールドのうちのいずれが有効であるかを知ることができる。

25 ロード／ストア命令の場合には、その内容を、IQ に書き込むと同時に、その書き込みが行われる IQ のエントリのアドレスと共に LSU に

送る。

③ 実行ステージ

I Qに保持されている未実行の命令は、データ駆動で処理される。従って、命令実行順序は **out-of-order** になる。以下に、命令の種類ごとに、
5 実行ステージにおける動作を説明する。

(a) 即値データのオペランド・スタックへのプッシュ命令

実行ステージにおける動作としては、何も行わない。

(b) 変数データのオペランド・スタックへのロード命令

10 I Qの、ロード命令を書き込みの内容とするエントリに関しては、同じ内容が命令デコード・設定ステージにおいてLSUに送られている。LSUでは、送られてきたロード命令を **out-of-order** で処理するようになっている。

I Qにおいて、オペランド・スタック上のデータをポップしてアドレス計算を行うようなロード命令 (Java VM における **iaload, laload, faload, daload, aaload, baload, caload, saload**) を書き込みの内容とし、有効なソース・フィールドに対応するWCFフィールドが全て1となっている (ソース・データが全てCRFに書き込み済みとなっている) エントリがあれば、そのエントリのアドレスと共にCRFをアクセスして得られるソース・データをLSUに送る。LSUは送られてきたソース・データをも
15 とにアドレス計算を行い、ロードの実行を試みる。

LSUで変数データが得られると、デスティネーションであるCRFのエントリのデータ・フィールドに変数データを書き込み、WCFフィールドを1に変更する。その上、I Qにおいて各ソース・フィールドで上記デスティネーションであるCRFのエントリのアドレスを照合し、
25 一致するソース・フィールドに対応するWCFフィールドを1とする。

この際、同じタイミングで命令が書き込まれるI Qのエントリにおいて

は、その書き込まれる内容と比較するようになっている。以上の動作が正常に終了すれば、そのロード命令を保持している I Q のエントリの S フィールドを正常終了に変更する。

(c) オペランド・スタック上のデータの変数へのストア命令

- 5 I Q において、ストア命令を書き込みの内容とし、有効なソース・フィールドに対応する WCF フィールドが全て 1 となっている（ソース・データが全て CRF に書き込み済みとなっている）エントリがあれば、そのエントリのアドレスと共に CRF をアクセスして得られるソース・データを LSU に送る。
- 10 オペランド・スタック上のデータをポップしてアドレス計算を行うようなストア命令（Java VM における iastore, lastore, fastore, dastore, aastore, bastore, castore, sastore）の場合、LSU は送られてきたソース・データをもとにアドレス計算を行う。
- 15 以上の動作が正常に終了すれば、そのストア命令を保持している I Q のエントリの S フィールドをストア実行可能に変更する。

正確な例外処理を保証するために、実際のストアの実行は完了ステージにおいて行う。

(d) 演算命令

- 20 I Q において、演算命令を書き込みの内容とし、有効なソース・フィールドに対応する WCF フィールドが全て 1 となっている（ソース・データが全て CRF に書き込み済みとなっている）エントリがあり、利用可能な状態の演算ユニットがあれば、そのエントリの内容をそのエントリのアドレス及び CRF をアクセスして得られるソース・データと共に利用可能な演算ユニットに送り実行させる。
- 25 演算の実行が正常に終了すれば、デスティネーションである CRF のエントリのデータ・フィールドに演算結果を書き込み、WCF フィールド

ドを1に変更する。その上、IQにおけるCRFのエントリのアドレスの照合及びWCFフィールドの変更を、上述のロード命令の場合と同様に行う。以上の動作が正常に終了すれば、その演算命令を保持しているIQのエントリのSフィールドを正常終了に変更する。

5 (e) オペランド・スタックの操作命令

スタック上でコピーを作成することのないオペランド・スタックの操作命令に関しては、実行ステージにおける動作としては、何も行わない。

本第1実施例においては、IQにおいて、スタック上でコピーを作成するようなオペランド・スタックの操作命令を書き込みの内容とし、有効なソース・フィールドに対応するWCFフィールドが全て1となっている（ソース・データが全てCRFに書き込み済みとなっている）エントリがあれば、その有効なソース・フィールドに示されるCRFのエントリからデータを読み出し、これを対応するデスティネーション・フィールドに示されるCRFのエントリのデータ・フィールドに書き込み、

10 WCFフィールドを1に変更する。その上、IQにおけるCRFのエントリのアドレスの照合及びWCFフィールドの変更を、上述のロード命令の場合と同様に行う。以上の動作が正常に終了すれば、そのオペランド・スタックの操作命令を保持しているIQのエントリのSフィールドを正常終了に変更する。

15

20 (f) 分岐命令

無条件分岐命令に関しては、実行ステージにおける動作としては、何も行わない。

IQにおいて、条件分岐命令を書き込みの内容とし、有効なソース・フィールドに対応するWCFフィールドが全て1となっている（ソース・データが全てCRFに書き込み済みとなっている）エントリがあれば、そのエントリの内容をそのエントリのアドレス及びCRFをアクセスし

25

て得られるソース・データと共に分岐ユニットに送り実行させる。

- 分岐ユニットにおいて条件式の計算が正常に終了すれば、その結果を分岐先アドレスと共に命令フェッチ・ユニットに通知する。以上の動作が正常に終了すれば、その条件分岐命令を保持している I Q のエントリ
- 5 の S フィールドを正常終了に変更する。

以上のように、I Q に保持されている未実行の命令は、実行可能となったものから処理されるので、命令実行順序は **out-of-order** になる。また、演算ユニット 0, 1、分岐ユニット及びロード／ストア・ユニットの各実行ユニットは互いに独立に並行して動作する。

- 10 ある命令の処理において例外事象が発生した場合には、その情報を、その命令を保持している I Q のエントリの S フィールドに書き込むと共に、命令フェッチ・ユニットに例外ベクタを通知する。

④ 完了ステージ

- ある命令が完了できるためには、プログラム上の順番でその命令より
- 15 も前にある命令が全て完了していなくてはならない。

I Q の完了ポインタで示されるエントリにおいて、S フィールドが実行済み／正常終了である、あるいはそうなりと、そのエントリに書き込まれている命令の内容に基づいて C P S 及び C R F を操作し、完了ポインタの値に 1 を加える。

- 20 C P S は、命令がデコード・設定された際の A P S の動作を再現すべく操作される。すなわち、ポップ／プッシュ操作を含む命令の場合には、有効なソース・フィールドの内容と同じものを順に C P S からポップし、有効なデスティネーション・フィールドの内容を順に C P S にプッシュする。スタック上でコピーを作成することのないオペランド・スタック
- 25 の操作命令の場合には、オペランド・スタックに対して行うべき操作を C P S に対して全く同様に行えばよい。本第 1 実施例においては、スタ

ック上でコピーを作成するようなオペランド・スタックの操作命令の場合には、有効なソース・フィールド及び有効なデスティネーション・フィールドを参照して、その命令のデコード・設定の際にAPSに対して行われた操作がCPSにおいて再現される。

- 5 本第1実施例においては、上述のCPSに対する操作に伴い、エントリ・アドレスがCPSからポップされるCRFのエントリでは、BBフィールドを0に変更し、そのエントリ・アドレスをFLに登録する。

- 10 IQの完了ポインタで示されるエントリにおいて、ストア命令が書き込まれている場合には、Sフィールドがストア実行可能である、あるいはそうなる、LSUに、上記ストア命令を保持しているIQのエントリのアドレスを示して、実際のストアの実行を依頼する。こうすれば、データが *in-order* でストアされることが保証できる。さらに、CPS及びCRFに対する操作を上と同様に行い、完了ポインタの値に1を加える。

- 15 以上のように、完了ポインタの値に1が加えられることによって、キューから除外されたIQのエントリに保持されていた命令は、完了したことになる。その命令よりも前にデコード・設定された命令はすべて完了しているので、命令の完了は *in-order* で行われることになる。

- 20 IQの完了ポインタで示されるエントリにおいて、Sフィールドが例外事象発生である、あるいはそうなった場合には、その時点におけるCPS及びCRFによって、プログラムが *in-order* で実行された場合の例外発生時点の状態が構成されるので、正確な例外処理が可能である。例外事象の発生した命令以降にデコード・設定された命令を全てキャンセルするには、キャンセルされるべき命令が書き込まれているIQのエン
- 25 トリの有効なデスティネーション・フィールドに示されるCRFのエントリの各々に対して、そのBBフィールドを0に戻し、そのエントリ・

アドレスをFLに登録することによって、割り付けを解除し、完了ポインタの値に1を加えたものを設定ポインタに書き込むことによって、キャンセルされるべき命令を保持しているIQのエントリを全てキューから除外すればよい。

- 5 以上が、本発明第1実施例の計算機システムの動作についての全般的な説明であるが、ついで、具体的な動作例について説明する。

いま、本第1実施例の計算機システムで、以下のようなプログラムを実行することを考えよう。

- dload [A] (変数名[A]に対応する倍精度浮動小数点データのロード)
- 10 dload [B] (変数名[B]に対応する倍精度浮動小数点データのロード)
- dadd (倍精度浮動小数点データ間の加算)
- d2f (倍精度浮動小数点データの単精度浮動小数点データへの変換)
- fload [T] (変数名[T]に対応する単精度浮動小数点データのロード)
- 15 dup_x1 (スタックトップの語のコピーを作成し、先頭から3語目に割り込ませる)
- fdiv (単精度浮動小数点データ間の除算)
- fsub (単精度浮動小数点データ間の減算)
- fstore [X] (スタックトップにある単精度浮動小数点データの変数名
- 20 [X]に対応する格納域へのストア)

以上のプログラムは、 $X = T - (A + B) / T$ の計算を行うものであるが、AとBのデータが倍精度で与えられ、この間の加算を倍精度のまま実行して、得られた加算データを単精度に変換して、以降は単精度で計算を行う、というものである。

- 25 第6図～第12図は、本第1実施例の計算機システムにおいて、上記プログラムを処理する際の動作をサイクル毎に示した説明図であり、以

下ではこの図をもとに詳細な動作を説明する。第6図～第12図において、CRF 6及びIQ 5の各エントリの構成は、それぞれ第3図、第5図のものと同じである。第6図～第12図で空白となっている箇所は、そのフィールドの内容に留意する必要が無いことを意味する。時系列で各構成要素の内容を示すために、各部の符号の後尾にハイフンと各サイクルに対応する数字を添えている。また、第6図～第12図において、APS、CPS、IQ及びCRFの各エントリは下から順に0、1、2、～のようにアドレスが付けられているものとする。

本動作例においては、説明を簡単にするため、変数データは全てデータ・キャッシュに格納され、CRFとデータ・キャッシュの間の Spill/Fill の動作は行わないものとする。従って、BP_OF_PS の値は終始0である。

また、本動作例においては、当初、APS、CPS、IQ及びCRFは初期化されており、FLにCRFの全てのエントリのアドレスが順に〈0〉、〈1〉、〈2〉、〈3〉……と書き込まれていて、この順で取り出されるものとする。

以下に、各サイクルにおける動作を、(A) 命令デコード・設定、(B) 実行及び (C) 完了の各ステージに分けて詳細に説明する。

(1-A) 第1サイクルの命令デコード・設定ステージ

命令 dload [A] のデコード・設定を行う。倍長語の変数データのオペランド・スタックへのロード命令であるので、FLに登録されているフリーなCRFの2エントリ6(0)、6(1)をそのデータを保持すべく割り付け、そのエントリのアドレス〈0〉、〈1〉をAPSにプッシュし、APSは3-1のようになる。

CRFの6(0)、6(1)の各エントリにおいては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込み、CRFは6-1のようになる。ここで、本動作例においては、終始分岐タグとして命

令デコード・設定ユニットから0が送られてくるものとする。

5 設定ポインタの値は0であるので、IQのエントリ5(0)に上記命令の内容を書き込み、IQは5-1のようになる。この際、APSにプッシュされるCRFのエントリのアドレス〈0〉,〈1〉を各々第1、第2デスティネーション・フィールドに書き込んでいる。さらに、設定ポインタの値に1を加え1にする。ここで、本動作例においては、IQのSフィールドには、命令が未実行であれば0、実行済み/正常終了あるいはストア命令におけるストア実行可能であれば1が書き込まれるものとする。

10 IQのエントリ5(0)に書き込まれるものと同じ上記命令の内容を、IQのエントリのアドレス0と共にLSUに送る。

(1-B) 第1サイクルの実行ステージ

当初のIQにおいては、実行可能な命令が書き込まれているエントリは存在しないので、実行ステージの動作としては何も行われぬ。

15 (1-C) 第1サイクルの完了ステージ

当初のIQの完了ポインタが示すエントリ5(0)において、命令はまだ書き込まれていないため、完了ステージの動作としては何も行われぬ。

(2-A) 第2サイクルの命令デコード・設定ステージ

20 命令 dload [B] のデコード・設定を行う。倍長語の変数データのオペランド・スタックへのロード命令であるので、FLに登録されているフリーなCRFの2エントリ6(2)、6(3)をそのデータを保持すべく割り付け、そのエントリのアドレス〈2〉,〈3〉をAPSにプッシュし、APSは3-2のようになる。

25 CRFの6(2)、6(3)の各エントリにおいては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込み、CRFは6-

2のようになる。

- 5 設定ポインタの値は1であるので、I Qのエントリ 5 (1)に上記命令の内容を書き込み、I Qは5-2のようになる。この際、APSにプッシュされるCRFのエントリのアドレス〈2〉, 〈3〉を各々第1、第2デスティネーション・フィールドに書き込んでいる。さらに、設定ポインタの値に1を加え2にする。

I Qのエントリ 5 (1)に書き込まれるものと同じ上記命令の内容を、I Qのエントリのアドレス1と共にLSUに送る。

(2-B) 第2サイクルの実行ステージ

- 10 LSUは変数Aのアクセスを開始する。レイテンシは2サイクルであるとする。

(2-C) 第2サイクルの完了ステージ

- 15 5-1の状態にあるI Qの完了ポインタが示すエントリ 5 (0)において、Sフィールドは0であるので、完了ステージの動作としては何も行われぬ。

(3-A) 第3サイクルの命令デコード・設定ステージ

- 20 命令 `dadd` のデコード・設定を行う。オペランド・スタックから4語のソース・データをポップして演算を行い、倍長語の演算結果をプッシュする演算命令であるので、APSから〈0〉, 〈1〉, 〈2〉, 〈3〉をポップし、FLに登録されているフリーなCRFの2エントリ 6 (4)、6 (5)を演算結果を保持すべく割り付け、そのエントリのアドレス〈4〉, 〈5〉をAPSにプッシュし、APSは3-3のようになる。

CRFの6 (4)、6 (5)の各エントリにおいては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

- 25 設定ポインタの値は2であるので、I Qのエントリ 5 (2)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリ

- のアドレス〈0〉, 〈1〉, 〈2〉, 〈3〉を各々第1～第4ソース・フィールドに、APSにプッシュされる〈4〉, 〈5〉を各々第1、第2デスティネーション・フィールドに書き込んでいる。また、6-2の状態にあるCRFの6(0)、6(1)、6(2)、6(3)の各エントリでWCFフィールドを読み出し、それぞれIQのWCF 1～4の各フィールドに送っている。さらに、設定ポインタの値に1を加え3にする。

(3-B) 第3サイクルの実行ステージ

LSUは変数Bのアクセスを開始する。レイテンシは2サイクルであるとする。

- 10 LSUから変数Aのデータを構成する2語 A_1、A_2 が送られてくるので、CRFのエントリ6(0)、6(1)のデータ・フィールドに各々書き込み、WCFフィールドを1に変更する。その上、IQにおいて各ソース・フィールドでCRFのエントリのアドレス〈0〉, 〈1〉を照合するが、この場合、同じタイミングで命令が書き込まれるIQのエントリ
- 15 5(2)の第1、第2ソース・フィールドで一致するので、同じエントリのWCF 1、2の各フィールドを1とする。(WCF 3、4の各フィールドについては、(3-A)で説明したように6-2の状態にあるCRFの対応するエントリのWCFフィールドが読み出され送られてくるので、これが書き込まれる。)
- 20 以上のようにIQのエントリ5(0)に書き込まれた命令の実行が正常に終了するので、5(0)のSフィールドを正常終了を意味する1に変更する。

(3-C) 第3サイクルの完了ステージ

- 5-2の状態にあるIQの完了ポインタが示すエントリ5(0)において、Sフィールドは0であるので、完了ステージの動作としては何も行
- 25 われない。

(4-A) 第4サイクルの命令デコード・設定ステージ

- 命令 d2f のデコード・設定を行う。オペランド・スタックから2語のソース・データをポップして変換演算を行い、1語の演算結果をプッシュする演算命令であるので、APSから〈4〉, 〈5〉をポップし、F
- 5 Lに登録されているフリーなCRFのエントリ6(6)を演算結果を保持すべく割り付け、そのエントリのアドレス〈6〉をAPSにプッシュし、APSは3-4のようになる。

CRFのエントリ6(6)においては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

- 10 設定ポインタの値は3であるので、IQのエントリ5(3)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリのアドレス〈4〉, 〈5〉を各々第1、第2ソース・フィールドに、APSにプッシュされる〈6〉を第1デスティネーション・フィールドに書き込んでいる。また、6-3の状態にあるCRFの6(4)、6(5)の各エ
- 15 ントリでWCFフィールドを読み出し、それぞれIQのWCF1、2の各フィールドに送っている。さらに、設定ポインタの値に1を加え4にする。

(4-B) 第4サイクルの実行ステージ

- LSUから変数Bのデータを構成する2語B_1、B_2が送られてくる
- 20 ので、CRFのエントリ6(2)、6(3)のデータ・フィールドに各々書き込み、WCFフィールドを1に変更する。その上、IQにおいて各ソース・フィールドでCRFのエントリのアドレス〈2〉, 〈3〉を照合するが、この場合、IQのエントリ5(2)の第3、第4ソース・フィールドで一致するので、同じエントリのWCF3、4の各フィールドを1とする。
- 25 以上のようにIQのエントリ5(1)に書き込まれた命令の実行が正常に終了するので、5(1)のSフィールドを正常終了を意味する1に変更

する。

(4-C) 第4サイクルの完了ステージ

5 5-3の状態にあるIQの完了ポインタが示すエントリ5(0)において、Sフィールドが1となったので、5(0)の内容に基づいてCPS(及びCRF)を操作する。すなわち、IQのエントリ5(0)のデスティネーション・フィールドに書き込まれている〈0〉,〈1〉をCPSにプッシュし、CPSは4-4のようになる。さらに、完了ポインタの値に1を加え1とし、これで、5(0)の命令は完了したことになる。

(5-A) 第5サイクルの命令デコード・設定ステージ

10 命令 fload [T] のデコード・設定を行う。1語の変数データのオペランド・スタックへのロード命令であるので、FLに登録されているフリーなCRFのエントリ6(7)をそのデータを保持すべく割り付け、そのエントリのアドレス〈7〉をAPSにプッシュし、APSは3-5のようになる。

15 CRFのエントリ6(7)においては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

設定ポインタの値は4であるので、IQのエントリ5(4)に上記命令の内容を書き込む。この際、APSにプッシュされるCRFのエントリ
20 のアドレス〈7〉を第1デスティネーション・フィールドに書き込んでいる。さらに、設定ポインタの値に1を加え5にする。

IQのエントリ5(4)に書き込まれるものと同じ上記命令の内容を、IQのエントリのアドレス4と共にLSUに送る。

(5-B) 第5サイクルの実行ステージ

25 5-4の状態にあるIQにおいて、演算命令が書き込まれているエントリ5(2)はWCFフィールドが全て1となっているので、このエントリ5(2)の内容をそのエントリのアドレス2及びCRFの6(0)、6(1)、

6 (2)、6 (3)の各エントリに書き込まれているソース・データと共に演算ユニット0に送り演算を開始させる。この演算のレイテンシは2サイクルであるとする。

(5-C) 第5サイクルの完了ステージ

- 5 5-4の状態にあるIQの完了ポインタが示すエントリ5 (1)において、Sフィールドが1となったので、5 (1)の内容に基づいてCPS (及びCRF) を操作する。すなわち、IQのエントリ5 (1)のデスティネーション・フィールドに書き込まれている〈2〉, 〈3〉をCPSにプッシュし、CPSは4-5のようになる。さらに、完了ポインタの値に1
- 10 を加え2とし、これで、5 (1)の命令は完了したことになる。

(6-A) 第6サイクルの命令デコード・設定ステージ

- 命令 `dup_x1` のデコード・設定を行う。命令 `dup_x1` は、ワード・スタックが、(右方向に成長するものとして), `word1`, `word2` のような状態であるとき、これを, `word2`, `word1`, `word2` と変えるような、スタック
- 15 上で1語のコピーを作成するオペランド・スタックの操作命令であるので、FLに登録されているフリーなCRFのエントリ6 (8)をコピー・データを保持すべく割り付け、3-5のように下から〈6〉, 〈7〉となっている状態のAPSを3-6のように〈8〉, 〈6〉, 〈7〉と変える。

- CRFのエントリ6 (8)においては、BBフィールドに1を立て、W
- 20 CF及びCの各フィールドには0を書き込む。

- 設定ポインタの値は5であるので、IQのエントリ5 (5)に上記命令の内容を書き込む。この際、コピー元となるデータを保持すべく割り付けられているCRFのエントリのアドレス〈7〉を第1ソース・フィールドに、コピー・データを保持すべく新たに割り付けられるCRFのエントリのアドレス〈8〉を第1デスティネーション・フィールドに書き
- 25 込んでいる。また、6-5の状態にあるCRFのエントリ6 (7)でWC

Fフィールドを読み出し、IQのWCF 1フィールドに送っている。さらに、設定ポインタの値に1を加え6にする。

(6-B) 第6サイクルの実行ステージ

5 LSUは変数Tのアクセスを開始する。レイテンシは2サイクルであるとする。

演算ユニット0で5(2)の演算命令の実行が正常に終了すれば、演算結果を構成する2語(A+B)₁、(A+B)₂が送られてくるので、CRFのエントリ6(4)、6(5)のデータ・フィールドに各々書き込み、WCFフィールドを1に変更する。その上、IQにおいて各ソース・フィールドでCRFのエントリのアドレス〈4〉、〈5〉を照合するが、この場合、IQのエントリ5(3)の第1、第2ソース・フィールドで一致するので、同じエントリのWCF 1、2の各フィールドを1とする。

15 以上のようにIQのエントリ5(2)に書き込まれた命令の実行が正常に終了するので、5(2)のSフィールドを正常終了を意味する1に変更する。

(6-C) 第6サイクルの完了ステージ

5-5の状態にあるIQの完了ポインタが示すエントリ5(2)において、Sフィールドは0であるので、完了ステージの動作としては何も行われない。

20 (7-A) 第7サイクルの命令デコード・設定ステージ

命令 `fdiv` のデコード・設定を行う。オペランド・スタックから2語のソース・データをポップして演算を行い、1語の演算結果をプッシュする演算命令であるので、APSから〈6〉、〈7〉をポップし、FLに登録されているフリーなCRFのエントリ6(9)を演算結果を保持すべく割り付け、そのエントリのアドレス〈9〉をAPSにプッシュし、APSは3-7のようになる。

CRFのエントリ 6 (9)においては、BBフィールドに1を立て、WCF及びCの各フィールドには0を書き込む。

- 5 設定ポインタの値は6であるので、IQのエントリ 5 (6)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリ
のアドレス〈6〉、〈7〉を各々第1、第2ソース・フィールドに、APS
にプッシュされる〈9〉を第1デスティネーション・フィールドに書
き込んでいる。また、6-6の状態にあるCRFの6 (6)、6 (7)の各エ
ントリでWCFフィールドを読み出し、それぞれIQのWCF 1、2の
各フィールドに送っている。さらに、設定ポインタの値に1を加え7に
10 する。

(7-B) 第7サイクルの実行ステージ

- 5-6の状態にあるIQにおいて、演算命令が書き込まれているエン
トリ 5 (3)は有効なWCFフィールドが全て1となっているので、この
エントリ 5 (3)の内容をそのエントリのアドレス3及びCRFの6 (4)、
15 6 (5)の各エントリに書き込まれているソース・データと共に演算ユニッ
ト0に送り演算を開始させる。この演算のレイテンシは2サイクルであ
るとする。

- LSUから変数Tのデータが送られてくるので、CRFのエントリ 6
(7)のデータ・フィールドに書き込み、WCFフィールドを1に変更する。
20 その上、IQにおいて各ソース・フィールドでCRFのエントリのアド
レス〈7〉を照合するが、この場合、IQのエントリ 5 (5)の第1ソー
ス・フィールド及び同じタイミングで命令が書き込まれる5 (6)の第2ソー
ス・フィールドで一致するので、5 (5)のWCF 1フィールド及び5 (6)
のWCF 2フィールドを1とする。(5 (6)のWCF 1フィールドについ
25 ては、(7-A)で説明したように6-6の状態にあるCRFのエント
リ 6 (6)のWCFフィールドが読み出され送られてくるので、これが書

き込まれる。)

以上のように I Q のエントリ 5 (4) に書き込まれた命令の実行が正常に終了するので、5 (4) の S フィールドを正常終了を意味する 1 に変更する。

5 (7-C) 第 7 サイクルの完了ステージ

5-6 の状態にある I Q の完了ポインタが示すエントリ 5 (2) において、S フィールドが 1 となったので、5 (2) の内容に基づいて C P S 及び C R F を操作する。すなわち、I Q のエントリ 5 (2) のソース・フィールドに書き込まれている〈0〉, 〈1〉, 〈2〉, 〈3〉を C P S からポップし、デスティネーション・フィールドに書き込まれている〈4〉, 〈5〉を C P S にプッシュし、C P S は 4-7 のようになる。エントリ・アドレスが C P S からポップされる C R F の 6 (0)、6 (1)、6 (2)、6 (3) の各エントリでは、B B フィールドを 0 に変更する。C R F のエントリのアドレス〈0〉, 〈1〉, 〈2〉, 〈3〉を F L に登録する。さらに、完了ポインタの値に 1 を加え 3 とし、これで、5 (2) の命令は完了したことになる。

(8-A) 第 8 サイクルの命令デコード・設定ステージ

命令 **fsub** のデコード・設定を行う。オペランド・スタックから 2 語のソース・データをポップして演算を行い、1 語の演算結果をプッシュする演算命令であるので、A P S から〈8〉, 〈9〉をポップし、F L に登録されているフリーな C R F のエントリ 6 (10) を演算結果を保持すべく割り付け、そのエントリのアドレス〈10〉を A P S にプッシュし、A P S は 3-8 のようになる。

C R F のエントリ 6 (10) においては、B B フィールドに 1 を立て、W C F 及び C の各フィールドには 0 を書き込む。

設定ポインタの値は 7 であるので、I Q のエントリ 5 (7) に上記命令

の内容を書き込む。この際、A P SからポップされるC R Fのエントリ
のアドレス〈8〉、〈9〉を各々第1、第2ソース・フィールドに、A P
Sにプッシュされる〈10〉を第1デスティネーション・フィールドに
書き込んでいる。また、6-7の状態にあるC R Fの6(8)、6(9)の各
5 エントリでW C Fフィールドを読み出し、それぞれI QのW C F 1、2
の各フィールドに送っている。さらに、設定ポインタの値に1を加え8
にする。

(8-B) 第8サイクルの実行ステージ

演算ユニット0で5(3)の変換演算命令の実行が正常に終了すれば、
10 1語の演算結果(A+B)が送られてくるので、C R Fのエントリ6(6)
のデータ・フィールドに書き込み、W C Fフィールドを1に変更する。
その上、I Qにおいて各ソース・フィールドでC R Fのエントリのアド
レス〈6〉を照合するが、この場合、I Qのエントリ5(6)の第1ソー
ス・フィールドで一致するので、同じエントリのW C F 1フィールドを
15 1とする。

以上のようにI Qのエントリ5(3)に書き込まれた命令の実行が正常
に終了するので、5(3)のSフィールドを正常終了を意味する1に変更
する。

5-7の状態にあるI Qにおいて、スタック上でコピーを作成するよ
うなオペランド・スタックの操作命令が書き込まれているエントリ5(5)
20 は有効なW C Fフィールドが1となっているので、データのコピーを実
行する。すなわち、C R Fのエントリ6(7)からデータを読み出し、こ
れをC R Fのエントリ6(8)のデータ・フィールドに書き込み、W C Fフ
ィールドを1に変更する。その上、I Qにおいて各ソース・フィールド
25 でC R Fのエントリのアドレス〈8〉を照合するが、この場合、同じタ
イミングで命令が書き込まれるI Qのエントリ5(7)の第1ソース・フ

ールドで一致するので、同じエントリのWCF 1フィールドを1とする。

- (WCF 2フィールドについては、(8-A)で説明したように6-7の状態にあるCRFのエントリ6(9)のWCFフィールドが読み出され送られてくるので、これが書き込まれる。)IQのエントリ5(5)のSフィールドを正常終了を意味する1に変更する。

(8-C) 第8サイクルの完了ステージ

5-7の状態にあるIQの完了ポインタが示すエントリ5(3)において、Sフィールドは0であるので、完了ステージの動作としては何も行われない。

- 10 (9-A) 第9サイクルの命令デコード・設定ステージ

命令 **fstore** [X] のデコード・設定を行う。スタックトップにある1語のデータのストア命令であるので、APSから〈10〉をポップし、APSは3-9のようになる。

- 15 設定ポインタの値は8であるので、IQのエントリ5(8)に上記命令の内容を書き込む。この際、APSからポップされるCRFのエントリのアドレス〈10〉を第1ソース・フィールドに書き込んでいる。また、6-8の状態にあるCRFのエントリ6(10)でWCFフィールドを読み出し、IQのWCF 1フィールドに送っている。さらに、設定ポインタの値に1を加え9にする。

- 20 IQのエントリ5(8)に書き込まれるものと同じ上記命令の内容を、IQのエントリのアドレス8と共にLSUに送る。

(9-B) 第9サイクルの実行ステージ

- 25 5-8の状態にあるIQにおいて、演算命令が書き込まれているエントリ5(6)は有効なWCFフィールドが全て1となっているので、このエントリ5(6)の内容をそのエントリのアドレス6及びCRFの6(6)、6(7)の各エントリに書き込まれているソース・データと共に演算ユニッ

ト 1 に送り演算を開始させる。この演算のレイテンシは 10 サイクルであるとする。

(9-C) 第 9 サイクルの完了ステージ

5 5-8 の状態にある I Q の完了ポインタが示すエントリ 5 (3) において、S フィールドが 1 となったので、5 (3) の内容に基づいて CPS 及び CRF を操作する。すなわち、I Q のエントリ 5 (3) のソース・フィールドに書き込まれている〈4〉、〈5〉を CPS からポップし、デスティネーション・フィールドに書き込まれている〈6〉を CPS にプッシュし、CPS は 4-9 のようになる。エントリ・アドレスが CPS から
10 ポップされる CRF の 6 (4)、6 (5) の各エントリでは、BB フィールドを 0 に変更する。CRF のエントリのアドレス〈4〉、〈5〉を FL に登録する。さらに、完了ポインタの値に 1 を加え 4 とし、これで、5 (3) の命令は完了したことになる。

(10-C) 第 10 サイクルの完了ステージ

15 5-9 の状態にある I Q の完了ポインタが示すエントリ 5 (4) において、S フィールドが 1 であるので、5 (4) の内容に基づいて CPS (及び CRF) を操作する。すなわち、I Q のエントリ 5 (4) のデスティネーション・フィールドに書き込まれている〈7〉を CPS にプッシュし、CPS は 4-10 のようになる。さらに、完了ポインタの値に 1 を加え
20 5 とし、これで、5 (4) の命令は完了したことになる。

(11-C) 第 11 サイクルの完了ステージ

5-10 の状態にある I Q の完了ポインタが示すエントリ 5 (5) において、S フィールドが 1 であるので、5 (5) の内容に基づいて CPS (及び CRF) を操作する。すなわち、(6-A) における APS の動作が
25 再現され、CPS は 4-11 のようになる。さらに、完了ポインタの値に 1 を加え 6 とし、これで、5 (5) の命令は完了したことになる。

(18-B) 第18サイクルの実行ステージ

演算ユニット1で5(6)の演算命令の実行が正常に終了すれば、1語の演算結果 $(A+B)/T$ が送られてくるので、CRFのエントリ6(9)のデータ・フィールドに書き込み、WCFフィールドを1に変更する。その上、IQにおいて各ソース・フィールドでCRFのエントリのアドレス〈9〉を照合するが、この場合、IQのエントリ5(7)の第2ソース・フィールドで一致するので、同じエントリのWCF2フィールドを1とする。

以上のようにIQのエントリ5(6)に書き込まれた命令の実行が正常に終了するので、5(6)のSフィールドを正常終了を意味する1に変更する。

(19-B) 第19サイクルの実行ステージ

5-18の状態にあるIQにおいて、演算命令が書き込まれているエントリ5(7)は有効なWCFフィールドが全て1となっているので、このエントリ5(7)の内容をそのエントリのアドレス7及びCRFの6(8)、6(9)の各エントリに書き込まれているソース・データと共に演算ユニット0に送り演算を開始させる。この演算のレイテンシは2サイクルであるとする。

(19-C) 第19サイクルの完了ステージ

5-18の状態にあるIQの完了ポインタが示すエントリ5(6)において、Sフィールドが1となったので、5(6)の内容に基づいてCPS及びCRFを操作する。すなわち、IQのエントリ5(6)のソース・フィールドに書き込まれている〈6〉、〈7〉をCPSからポップし、デスティネーション・フィールドに書き込まれている〈9〉をCPSにプッシュし、CPSは4-19のようになる。エントリ・アドレスがCPSからポップされるCRFの6(6)、6(7)の各エントリでは、BBフィー

ルドを0に変更する。CRFのエントリのアドレス〈6〉、〈7〉をFLに登録する。さらに、完了ポインタの値に1を加え7とし、これで、5(6)の命令は完了したことになる。

(20-B) 第20サイクルの実行ステージ

- 5 演算ユニット0で5(7)の演算命令の実行が正常に終了すれば、1語の演算結果 $T-(A+B)/T$ が送られてくるので、CRFのエントリ6(10)のデータ・フィールドに書き込み、WCFフィールドを1に変更する。その上、IQにおいて各ソース・フィールドでCRFのエントリのアドレス〈10〉を照合するが、この場合、IQのエントリ5(8)の第1ソース・フィールドで一致するので、同じエントリのWCF1フィールドを1とする。
- 10

以上のようにIQのエントリ5(7)に書き込まれた命令の実行が正常に終了するので、5(7)のSフィールドを正常終了を意味する1に変更する。

15 (21-B) 第21サイクルの実行ステージ

- 5-20の状態にあるIQにおいて、ストア命令が書き込まれているエントリ5(8)は有効なWCFフィールドが1となっているので、IQのエントリのアドレス8と共にCRFのエントリ6(10)に書き込まれているソース・データをLSUに送る。5(8)のSフィールドをストア実行可能を意味する1に変更する。
- 20

(21-C) 第21サイクルの完了ステージ

- 5-20の状態にあるIQの完了ポインタが示すエントリ5(7)において、Sフィールドが1となったので、5(7)の内容に基づいてCPS及びCRFを操作する。すなわち、IQのエントリ5(7)のソース・フィールドに書き込まれている〈8〉、〈9〉をCPSからポップし、デスティネーション・フィールドに書き込まれている〈10〉をCPSにプ
- 25

ツッシュし、CPSは4-21のようになる。エントリ・アドレスがCPSからポップされるCRFの6(8)、6(9)の各エントリでは、BBフィールドを0に変更する。CRFのエントリのアドレス〈8〉、〈9〉をFLに登録する。さらに、完了ポインタの値に1を加え8とし、これで、

5 5(7)の命令は完了したことになる。

(22-C) 第21サイクルの完了ステージ

5-21の状態にあるIQの完了ポインタが示すエントリ5(8)においては、ストア命令が書き込まれており、Sフィールドが1となったので、LSUにIQのエントリのアドレス8を示して、データ・キャッシュへのストアの実行を依頼する。さらに、5(8)の内容に基づいてCPS及びCRFを操作する。すなわち、IQのエントリ5(8)のソース・フィールドに書き込まれている〈10〉をCPSからポップし、CPSは4-22のようになる。エントリ・アドレスがCPSからポップされるCRFのエントリ6(10)では、BBフィールドを0に変更する。CRF

10 のエントリのアドレス〈10〉をFLに登録する。さらに、完了ポインタの値に1を加え9とし、これで、5(8)の命令は完了したことになる。

15

以上で、本第1実施例の計算機システムにおいて $X=T-(A+B)/T$ の計算が完了したことになる。

本発明の計算機システムにおいては、分岐予測に基づく投機的実行を実現することができる。APS履歴ファイルは、投機的実行を可能にするために具備されるものである。条件分岐命令がデコードされるごとに、APS履歴ファイルの1エントリにAPSの全エントリ及びPP_OF_APSの内容を書き込むようになっている。以下に、本実施例の計算機システムにおいて、分岐予測に基づく投機的実行がどのように行

20

25 われるかについて説明する。

前述のように、本実施例の計算機システムにおいては、命令デコード

・設定ステージにおいて、命令をデコードして、命令の内容に応じてA
P S及びC R Fを操作すると共に、命令の内容をI Qに書き込むよう
になっている。初期状態から命令が流れ始め最初の条件分岐命令がデコー
ドされるまでの間、デコードされる命令に分岐タグとして0を付し、こ
5 の分岐タグ0を、命令の内容が書き込まれるI QのエントリのB Tフイ
ールド、及び、割り付けられるC R FのエントリのCフィールドに書き
込む。

最初の条件分岐命令がデコードされ分岐予測が行われる際に、分岐時
点の状態を保存するために、A P Sの全エントリ及び P P _ O F _ A P S の内
10 容をA P S履歴ファイルのアドレス0のエントリに書き込む。上記の分
岐予測に基づいた命令の流れにおいては、分岐タグとして1を付し、I
Q及びC R Fの設定を行う。

2つ目の条件分岐命令がデコードされた時に、最初の条件分岐命令が
未確定である場合、あるいは確定して予測が当たっていた場合には、A
15 P Sの全エントリ及び P P _ O F _ A P S の内容をA P S履歴ファイルのアド
レス1のエントリに書き込む。2段目の分岐予測に基づいた命令の流れ
においては、分岐タグとして2を付し、I Q及びC R Fの設定を行う。

分岐予測が当たり続ければ以後同様に処理が進み、A P S履歴ファイ
ルへの書き込みはアドレス順に行われる。また、A P S履歴ファイルの
20 アドレス n のエントリに書き込みが行われてから次に書き込みが行わ
れるまでの間にデコードされる命令には分岐タグとして n+1 を付すも
のとする。

分岐予測が外れた場合には、その条件分岐命令以降にデコードされた
命令に付された分岐タグをもとに、演算ユニット、分岐ユニット及びL
25 S Uの各実行ユニットに送られた命令をキャンセルし、C R Fにおいて
Cフィールドで分岐タグを照合してその一致するエントリの各々に対し

て、そのBBフィールドを0に変更して、そのエントリのアドレスをFLに登録し、IQの設定ポインタの値をその条件分岐命令が書き込まれているエントリの次のアドレスに書き換えることによって、その条件分岐命令以降にデコード・設定された命令を無効とする。さらに、同じエントリ・アドレスにあるCPSのエントリとその内容が一致しないAPSの各エントリ及びPP_OF_APSに、その条件分岐命令がデコードされた際にAPS履歴ファイルに書き込まれた内容をコピーして、正しい位置の命令から処理を再開する。

10 以上のように、本発明の計算機システムにおいては、APS履歴ファイルを用いることによって、条件分岐命令がデコードされ分岐予測が行われる各々の時点の状態を再構成することができるので、分岐予測に基づく投機的実行が可能である。

15 以上では、説明を簡単にするため、1サイクルで同時にデコード・設定／完了できる命令は高々1つまでとして説明してきた。本発明の計算機システムにおいては、同時に複数の命令をデコード・設定／完了できる構成とすることができる。すなわち、FLがFIFOキューの構成となっていれば、割り付けのためにフリーなCRFのエントリのアドレスをFLから取り出す順番は決まっており、各命令における何語ポップし何語プッシュするかというようなスタック操作の内容を把握して、同時に20 複数の命令をデコード・設定することができる。また、命令の完了の動作においては、各命令におけるCPSに対するスタック操作があらかじめ厳密に決められているので、より容易に複数命令の同時完了を実現することができる。

25 同時にデコード・設定／完了できる命令の数を多くするほど、命令デコード・設定ユニットその他の制御回路が複雑になると共に、IQやCRFを構成する各レジスタ・ファイルのポートの数やIQの各エントリ

のソース・フィールドごとに設けられる比較回路の数、演算ユニットの数、さらに構成要素間を結合するバスの数などの点で、より多量のハードウェアが必要となる。

5 本発明の計算機システムにおいては、デコード・設定を2つのステージに分けて行うこととし、その前半のステージにおいて、同時にデコード・設定する複数の命令の内容を統合した形式に変換するような構成とすることも可能である。

たとえば、1サイクル当り3命令までデコード・設定できるような構成をとる場合、前述の $X=T-(A+B)/T$ を計算するプログラムは第13図
10 の図表に示されるような内容に変換される。第13図の図表の各段には、同時にデコード・設定される3つの命令に基づく、PP_OF_APSの増分、APSの操作内容及びIQの3エントリに書き込まれるべき設定内容を示している。ここでは、設定前のAPSの内容を.....s2, s1, s0（右端がスタックトップ）、FIFOキューの構成となっているフリー・リストの
15 内容を（取り出される順に）f1, f2, f3として記述しており、デコード・設定の後半のステージにおいて、それぞれ対応するCRFのエントリ・アドレスがAPS/IQに書き込まれるようになっている。PP_OF_APSの増分の欄で示されるようにAPSのスタックトップの位置が移動するが、APSの操作内容の欄では、この移動後のスタック
20 ップの位置が右端に対応している。また、'NC'は「変化なし(No Change)」を意味する。

ついで、本発明第2実施例の計算機システムについて説明する。

25 第2実施例は、オペランド・スタックの操作命令の処理方法が、第1実施例と異なる。

第2実施例の計算機システムは、第1実施例とは、統合レジスタ・フ

ファイル (CRF) 6 のエントリの構成が異なるが、計算機システムの基本構成、及び、前進ポインタ・スタック (APS) 3、完了ポインタ・スタック (CPS) 4、命令キュー (IQ) 5 の構成は同様である。

第 14 図は、本第 2 実施例における、CRF 6 の各々のエントリ 6 (i) の詳細な構成を示す説明図である。ここで、i はエントリのアドレスである。CRF 6 の各々のエントリ 6 (i) はデータ・フィールド 6 1 (i)、書込み完了フラグ (WCF, Write Completion Flag) フィールド 6 2 (i)、カラー (C, Colour) フィールド 6 3 (i)、ビジービット (BB) フィールド 6 4 (i)、及び参照数 (NR, Number of Reference) フィールド 6 5 (i) から成っている。

CRF の各々のエントリの、データ・フィールド及び WCF、C、BB の各フィールドは第 1 実施例と同様である。

CRF の各々のエントリにおいて、NR フィールドは、その CRF のエントリのアドレスを保持している CPS のエントリの数が書き込まれているようになっている。

すなわち、CRF とデータ・キャッシュの間の Spill/Fill の動作及び命令の完了に基づく CPS に対する操作に伴い、CRF の関係するエントリにおいて、そのエントリのアドレスを保持する CPS のエントリの数の更新を行うようになっている。

第 2 実施例においては、CRF からデータ・キャッシュへの 1 語のデータのストア (Spill) は以下のように行われる。この場合、BP_OF_PS で示されるエントリの内容が APS と CPS で一致していなければならないが、その 2 つのポインタ・スタックで一致する BP_OF_PS で示されるエントリの内容で示される CRF のエントリに書き込まれている 1 語分のデータをデータ・キャッシュにストア (Spill) し、BP_OF_PS の値に 1 を加える。さらに、上記 CRF のエントリにおいて、NR フィールド

ドの値から1を引く。その結果、その値が0になれば、そのCRFのエントリのBBフィールドを0に変更し、そのエントリのアドレスをFLに登録する。

- 5 逆に、データ・キャッシュからCRFへの1語のデータのロード (Fill) は以下のように行われる。すなわち、最後にストア (Spill) した1語分のデータをデータ・キャッシュから取り出し、それにFLに登録されているフリーなCRFの1エントリを割り付け、そのデータ・フィールドに書き込む。WCF、NR、BBの各フィールドは1とする。さらに、その割り付けられたCRFのエントリのアドレスを、APS及びCPS
10 のBP_OF_PSに示されるエントリの1つ下に各々書き込み、BP_OF_PSの値から1を引く。

命令の処理においても、本第2実施例の計算機システムは第1実施例と概ね同様であるが、以下に、各ステージごとに第1実施例との相違を明確にすることにより、本第2実施例の計算機システムの動作を説明す
15 る。

① 命令フェッチ・ステージ

第1実施例と同様。

② 命令デコード・設定ステージ

以下の点を除き、第1実施例と同様。

- 20 ・命令のデコード・設定に伴い新たに割り付けられるCRFのエントリにおいて、NRフィールドに0を書き込む。
- ・オペランド・スタックの操作命令の場合、オペランド・スタックに対して行うべき操作をAPSに対して全く同様に行う。その命令の内容をIQに書き込む際に、Sフィールドは実行済みとする。また、スタック上
25 でコピーを作成するようなオペランド・スタックの操作命令 (Java VMにおける dup, dup2, dup_x1, dup2_x1, dup_x2 及び dup2_x2) の場合にも、

- FLに登録されているフリーなCRFのエントリを新たに割り付けることはしないので、ソース・フィールド及びデスライネーション・フィールドへの書き込みは必要ない。
- ③ 実行スライジ
- 以下の点を除き、第1実施例と同様。
- ・オペランド・スタックの操作命令に関しては、スタック上でコピーを作成するような命令の場合も、実行スライジにおける動作としては、何も行わない。
 - ④ 完了スライジ
- 以下の点を除き、第1実施例と同様。
- 10
- ・オペランド・スタックの操作命令の完了においては、スタック上でコピーを作成するような命令の場合も、オペランド・スタックに対して行うべき操作をCRPに対して全く同様に行う（命令がデコード・設定された際のAPRの動作を再現すべくCRPが操作される、という観点からは第1実施例と同様）。
- 15
- ・命令の完了に基づくCRPに対する操作に伴い、関係するCRFのエントリのNRフィールドの値を増減させる。すなわち、エントリ・アドレスがCRPにプッシュされるCRFのエントリではNRフィールドの値を0から1に変更し、スタック上でコピーを作成するようなオペレーション・スタックの操作命令の完了に伴いエントリ・アドレスがCRP上でコピーされるCRFのエントリではNRフィールドの値に1を加え、エントリ・アドレスがCRPからポップされるCRFのエントリではNRフィールドの完了に基づくCRPに対する操作に伴い、エントリ・アドレスがCRPからポップされるCRFのエントリでは、NRフィールドの値が0になった場合にはのみ、BBフィールドを0に変更し、そのエントリ
- 20
- 25

・アドレスをFLに登録する。

以上が、本第2実施例の計算機システムの動作についての全般的な説

明である。

第15図～第21図は、本第2実施例の計算機システムにおいて、前

述の $X = T \cdot (A + B) / T$ の計算を行うプログラムの処理する際の動作をサイ

クル毎に示した説明図であり、図中CRF6及びIQ5の各エントリの

構成は、それぞれ第14図、第5図のものと同じである。第15図～第

21図で空白となっている箇所は、そのフィールドの内容に留意する必

要が無いことを意味する。時系列で各構成要素の内容を示すために、各

部の符号の後尾にハイフンと各サイクルに対応する数字、及び、第1実

施例における動作例を示す第6図～第12図における符号と区別するた

めに文字 'a' を添えている。

本発明の計算機システムは、上述の実施例に限られるものではなく、

細部の構成の異なる様々な実施例が存在する。たとえば、次のようなも

のをあげることができる。

① IQが各々のエントリにソース・データも書き込まれるような構成と

なっており、IQにおいて、対応するWCFフィールドに1が立つのと

同じタイミングでソース・データを書き込むようにしたもの。

② 各実行ユニットの入力段にリザーブ・スレーションを設けた構

成とし、命令デコード・設定スレージにおいて、個々の命令の内容をI

Qに書き込むと共にそれぞれ適切なリザーブ・スレーション・スレーションへ送

るようにしたもの (LSUに関しては、リザーブ・スレーション・スレーション

と共にストア・バッファを備えた構成としてもよい)。

③ 整数データ用/浮動小数点データ用、あるいは32ビット・データ用

/64ビット・データ用というようにデータ型別にCRF及びブリ

ストを備え、さらに、それぞれに対応してひと揃いの実行ユニットを設けたもの。

これらの実施例の多くは、レジスタ・ベースのスーパーカラム・アーキテクチャにおいて同様な主旨のものを見出すことのできるものである。

命令セットがスタック型の命令及びレジスタ型の命令を共に含むよう

な、本発明に基づく計算機システムも実現可能である。すなわち、前進ポインタ・スタック及び完了ポインタ・スタックに加えて、レジスタ番号にそれぞれ対応して設けられた各エントリに統合レジスタ・ファイルのエントリ・アドレスが書き込まれるようになっている前進レジスタ・マップ

10 ピング・チャネル及び完了レジスタ・マップ・チャネルを具備する構

成とし、スタック型の命令に関しては前進／完了ポインタ・スタックを操作し、レジスタ型の命令に関しては前進／完了レジスタ・マップ・チャネルをアクセスするようにする。この場合、前進ポインタ・スタック履歴ファイルの代わりに、各々のエントリに前進ポインタ・スタック及び前進レジスタ・マップ・チャネル双方の内容が書き込まれるよう

15 になっている前進履歴ファイルを具備する必要がある。

産業上の利用可能性

20 以上のように、本発明の計算機システムは、正確な例外処理を保証しつつ、スタックマシンの機械語で記述されたプログラムを out-of-order で処理するものであるが、複数の演算ユニットによる並列処理やそれらのパイプライン化によって効率的な処理を行うことが可能であるという利点がある。

25 また、分岐予測に基づく投機的実行や、1 サイクル当り複数命令のデコード・設定／完了の可能な構成とすることにより、さらなる高速化が可能である。

請 求 の 範 囲

50

1. データ・キャッシュ (11) と
 各々のエントリにデータが書き込まれるようになっている統合レジスタ・ファイル (6) と
- 5 各々のエントリに上記統合レジスタ・ファイル (6) のエントリのアドレスが書き込まれるようになっているスタックの構成となっている前進ポインタ・スタック (3) と
 各々のエントリに個々の命令の内容が書き込まれるようになっている F I F O (First In First Out) キューの構成となっている命令キュー (5) と
- 10 演算を実行するようになっている演算ユニット (80, 81) と
 上記データ・キャッシュ (11) 及び上記統合レジスタ・ファイル (6) にアクセスできるようになっているロード/ストア・ユニット (83) とを具備し、
- 15 オペランド・スタックに対するポインタ操作を含む命令がデコードされた場合には、ポインタすべき語数と同じ数だけ上記統合レジスタ・ファイル (6) のエントリのアドレスを上記前進ポインタ・スタック (3) からポインタし、
- 20 オペランド・スタックに対するポインタ操作を含む命令がデコードされた場合には、ポインタすべき語数と同じ数だけ割り付けられていない上記統合レジスタ・ファイル (6) のエントリを割り付け、上記割り付けられた上記統合レジスタ・スタック (3) にポインタし、
- 25 デコードされた命令の内容を、ポインタ/ポインタ操作を伴う命令の場合にはポインタ/ポインタされる上記統合レジスタ・ファイル (6) のエ

ントリのアドレスと共に、上記命令キュー (5) に書き込み、
 上記命令キュー (5) に保持されている未実行の命令をデータ駆動の
 原理に基づき処理するようになっている、スタックマシンの機械語で記
 述されたプログラムを実行する計算機システム。

5 2. 各々のエントリに上記統合レジスタ・フাইル (6) のエントリの
 アドレスが書き込まれるようになっているスタックの構成となっている
 完了ポインタ・スタック (4) を具備し、

10 上記命令キュー (5) の先頭のエントリにおいて保持されている命令
 の完了が可能である、あるいはそうなる、上記命令キュー (5) の上
 記先頭のエントリの内容に基づき、上記保持されている命令がデコード
 された際の上記前進ポインタ・スタック (3) の動作を再現すべく上記
 完了ポインタ・スタック (4) を操作し、上記命令キュー (5) から上
 記先頭のエントリを除外し、

15 ポツポツ操作によって上記完了ポインタ・スタック (4) におけるアド
 レスの保持が無くなった上記統合レジスタ・フাইル (6) のエントリ
 の割り付けを解除するようになっている請求項 1 記載の計算機シス
 ム。

3. 割り付けられていない上記統合レジスタ・フাইル (6) のエント
 リのアドレスを保持するフリー・リストを具備し、

20 初期状態においては、上記統合レジスタ・フাইル (6) の全てのエ
 ントリのアドレスが上記フリー・リストに登録されており、

上記統合レジスタ・フাইル (6) のエントリを割り付ける必要があ
 る場合に、上記フリー・リストから割り付けられていない上記統合レジ
 スタ・フাইル (6) のエントリのアドレスを取り出し、

25 割り付けが解除された上記統合レジスタ・フাইル (6) のエントリ
 のアドレスを上記フリー・リストに登録するようになっている請求項 2

記載の計算機システム。

4. 上記前進ボイソタ・スタック(3)と上記完了ボイソタ・スタック(4)

が循環型のバッファの構成となっており、

上記前進ボイソタ・スタック(3)及び上記完了ボイソタ・スタック

(4)で、上記統合レジスタ・フাইル(6)のエントリのアドレスを

保持する最下位のエントリの内容が一致する場合には、上記前進ボイソ

タ・スタック(3)及び上記完了ボイソタ・スタック(4)において上記

最下位のエントリにおける上記統合レジスタ・フাইル(6)のエント

リのアドレスの保持を解除し、上記一致する内容で示される上記統合レ

ジスタ・フাইル(6)のエントリに書き込まれているデータを上記デ

ータ・キャッシュ(11)にストア(Spill)することができるようにな

っており、

上記データ・キャッシュ(11)に最後にストア(Spill)したデータ

に対し、割り付けられていない上記統合レジスタ・フাইル(6)の1

エントリを割り付け、上記データを書き込み、上記前進ボイソタ・スタ

ック(3)及び上記完了ボイソタ・スタック(4)において、上記統合

レジスタ・フাইル(6)のエントリのアドレスを保持する最下位のエ

ントリの1つ下のエントリに上記データが書き込まれる上記統合レジ

スタ・フাইル(6)のエントリのアドレスを保持させることによって、

上記最後にストア(Spill)したデータの上記統合レジスタ・フাইル(6)

へのロード(Fill)を行えるようになっている請求項2記載の計算機シ

ステム。

5. 各々のエントリに上記前進ボイソタ・スタック(3)の内容が書き

込まれるようになっている上記前進ボイソタ・スタック履歴フাইル(3a)

を具備し、

上記統合レジスタ・フাইル(6)が各々のエントリに分岐タグも書

き込まれるような構成となっており、
命令のデコードの際に、割り付けられる上記統合レジスタ・ファイル
(6)のエントリに分岐タグを書き込むようになっており、
条件分岐命令がデコードされるごとに、上記前進ポインタ・スタック
履歴ファイル(3a)の1エントリに上記前進ポインタ・スタック(3)

の内容を書き込み、分岐タグを変更して、分岐予測に基づく投機的実行
を行い、

分岐予測が外れた場合には、その条件分岐命令以降にデコードされた

命令を無効にし、上記条件分岐命令以降にデコードされた命令に付した
分岐タグが書き込まれている上記統合レジスタ・ファイル(6)のエン

トリの割り付けを解除し、上記条件分岐命令がデコードされた際に書き
込みの行われた上記前進ポインタ・スタック履歴ファイル(3a)のエ
ントリの内容を、上記前進ポインタ・スタック(3)にコピーして、正
しい位置の命令から処理を再開することによって、

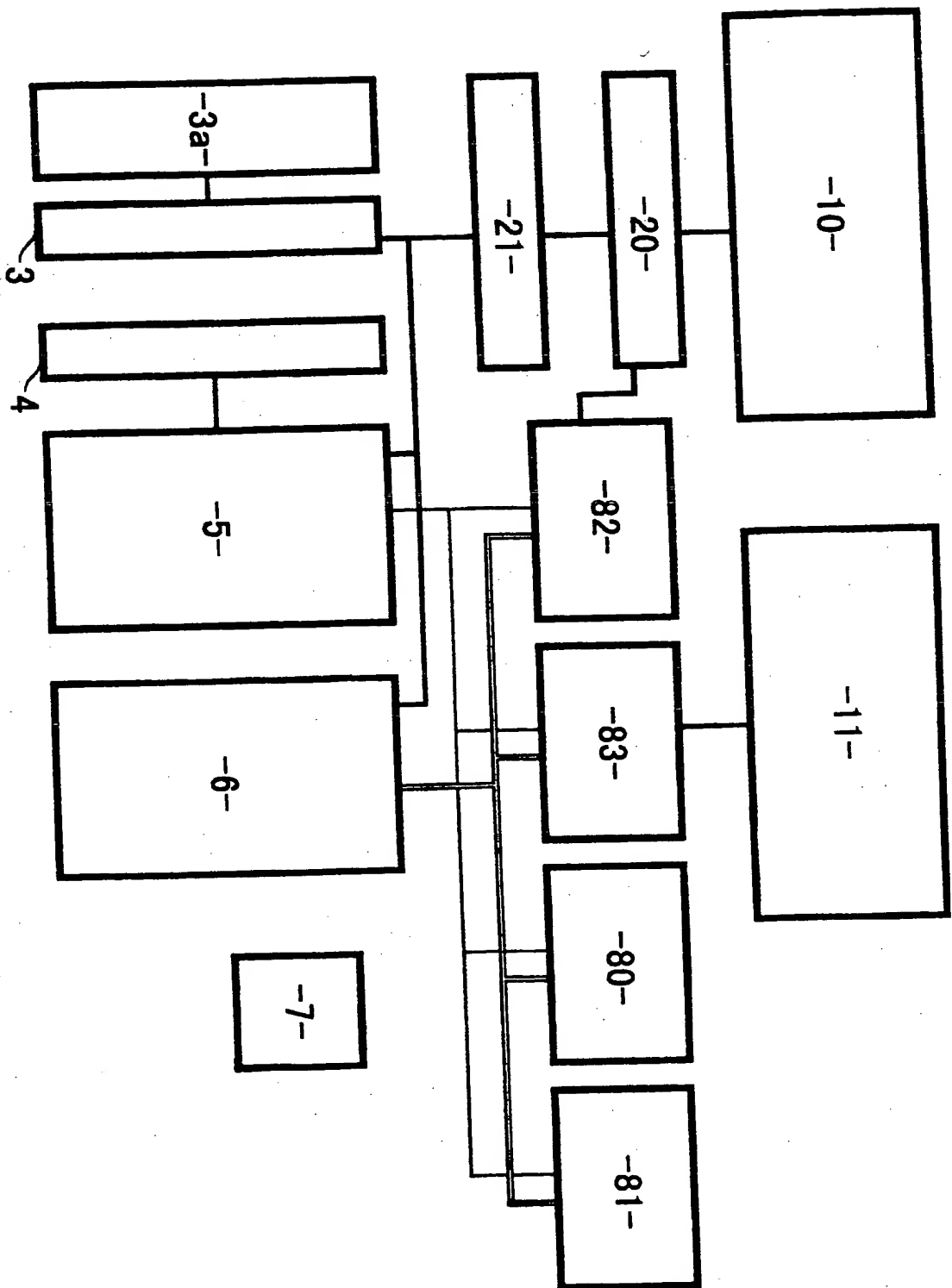
分岐予測に基づく投機的実行を行うようになっていて、請求項2記載の
計算機システム。

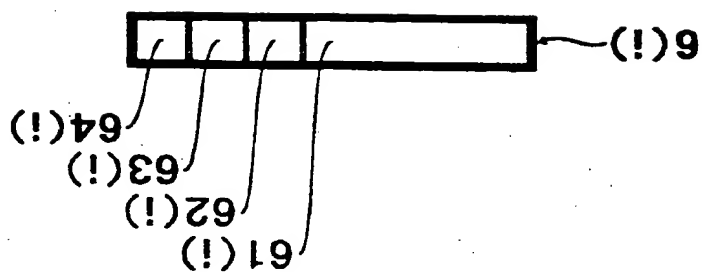
6. 上記フリー・リストがFIFOキューの構成となっており、

同時に複数の命令をデコードし、上記前進ポインタ・スタック(3)
の操作、上記統合レジスタ・ファイル(6)のエントリの割り付け及び
上記命令キュー(5)の連続する複数のエントリへの命令の内容の書き
込みを行う機能と、

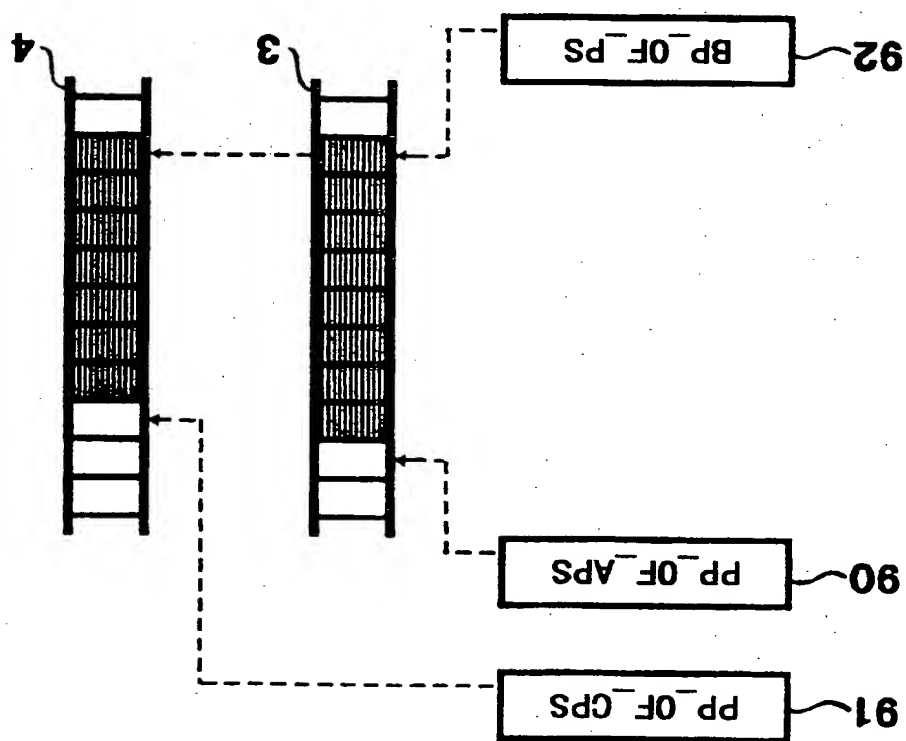
同時に上記命令キュー(5)の連続する複数のエントリに書き込まれ
ている内容に基づき、上記完了ポインタ・スタック(4)の操作及び上
記統合レジスタ・ファイル(6)のエントリの割り付けの解除を行う機
能を有する請求項3記載の計算機システム。

第 1 图





第 3 図



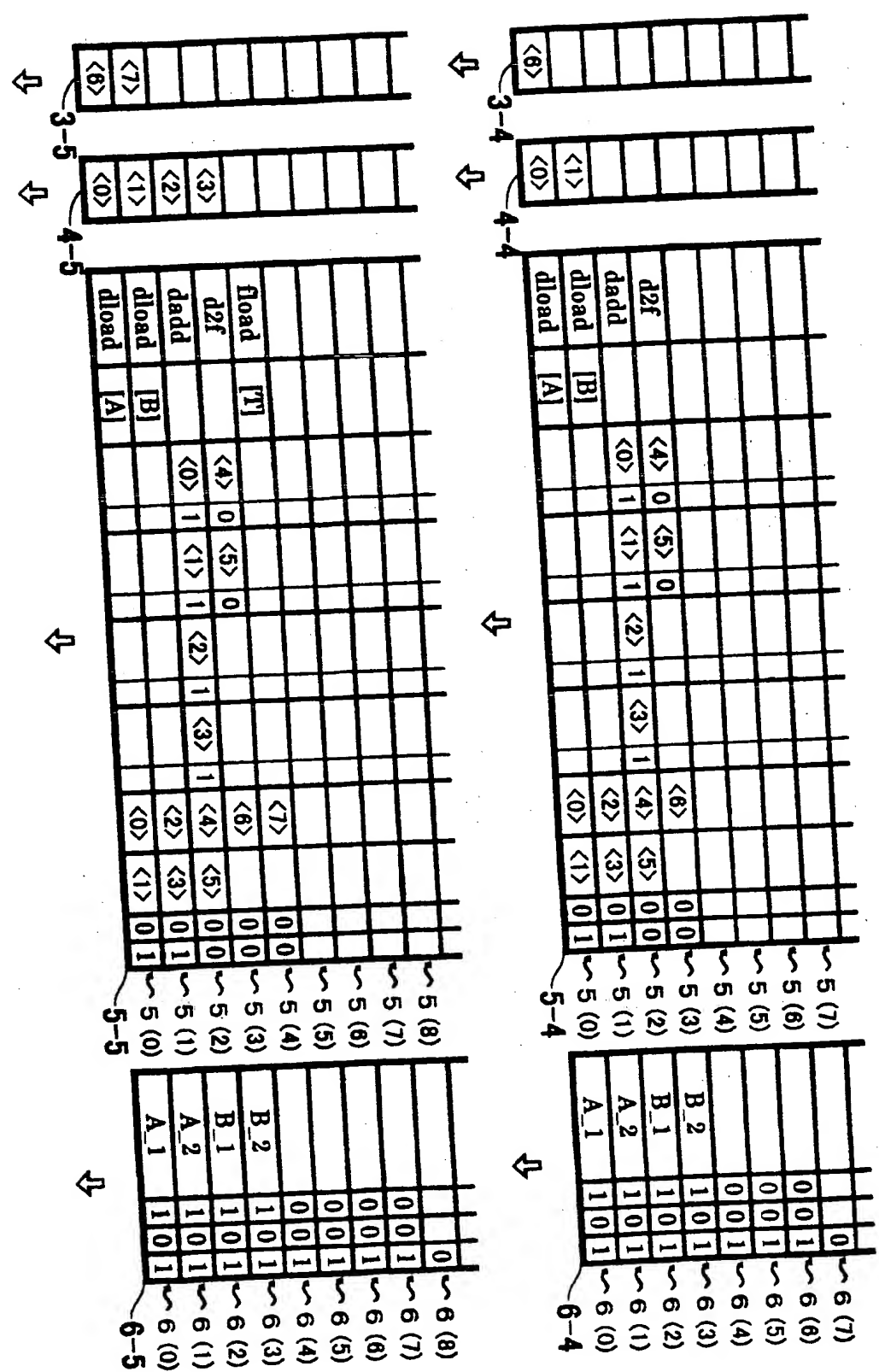
第 2 図



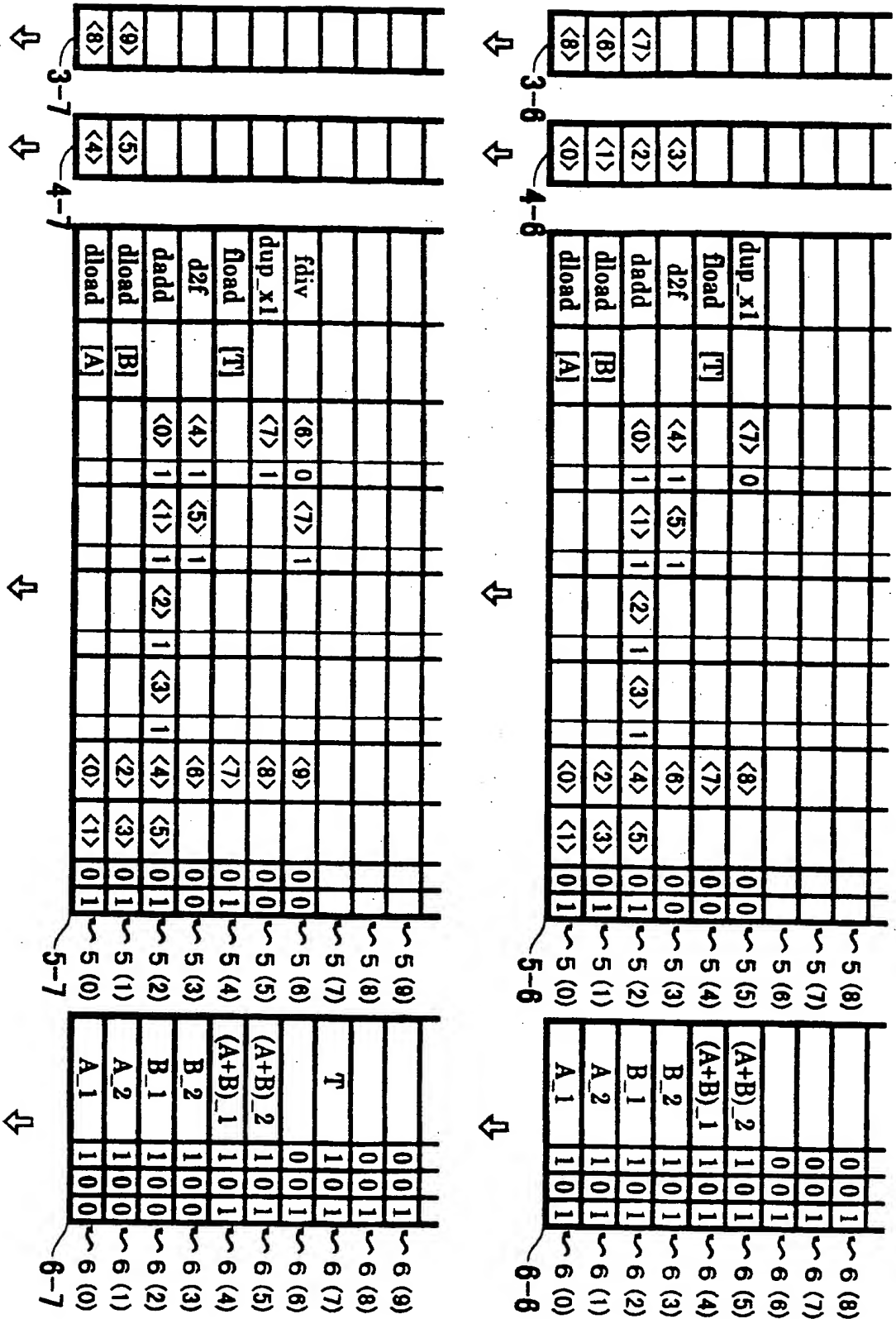
4/18



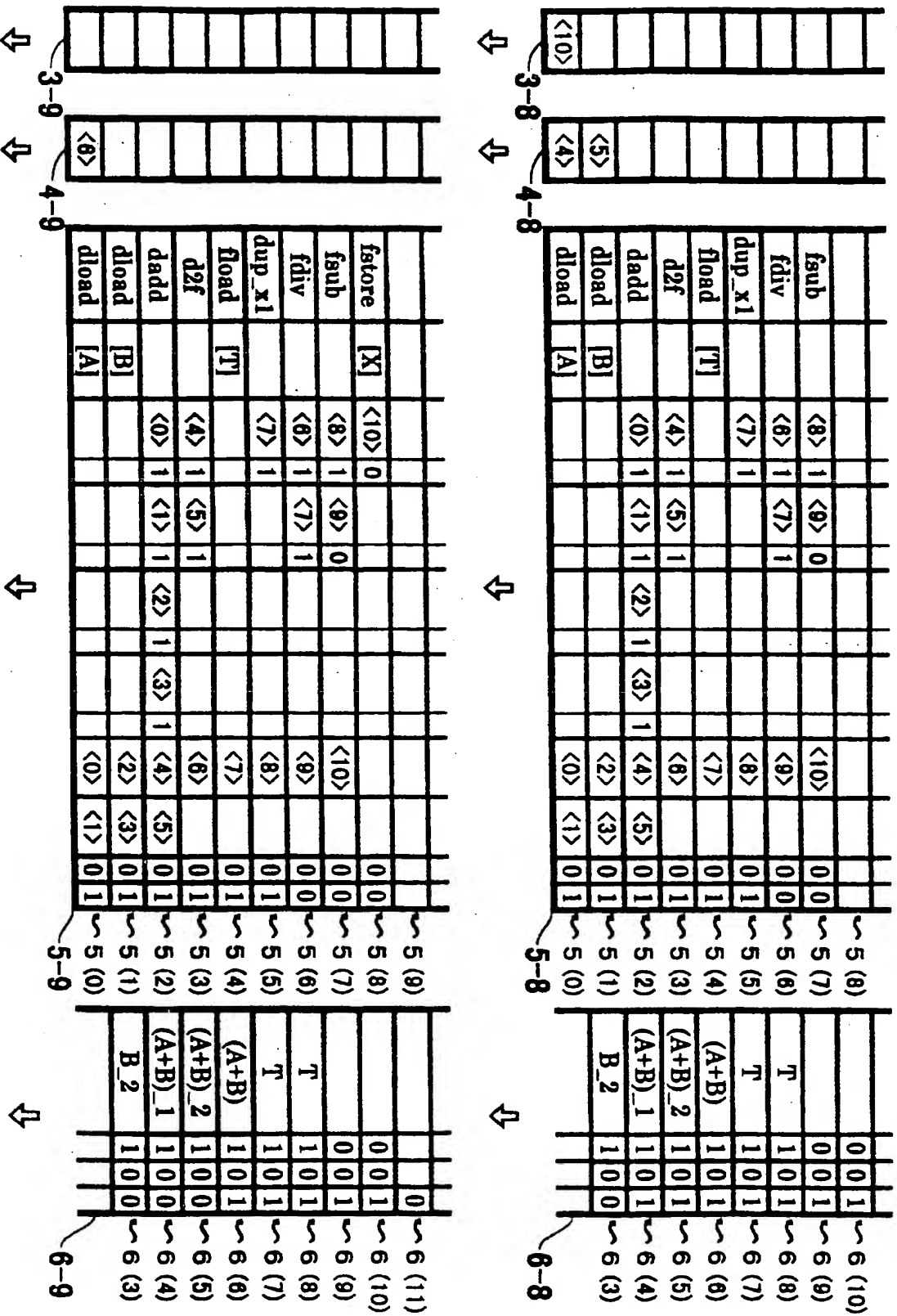
第 7 図



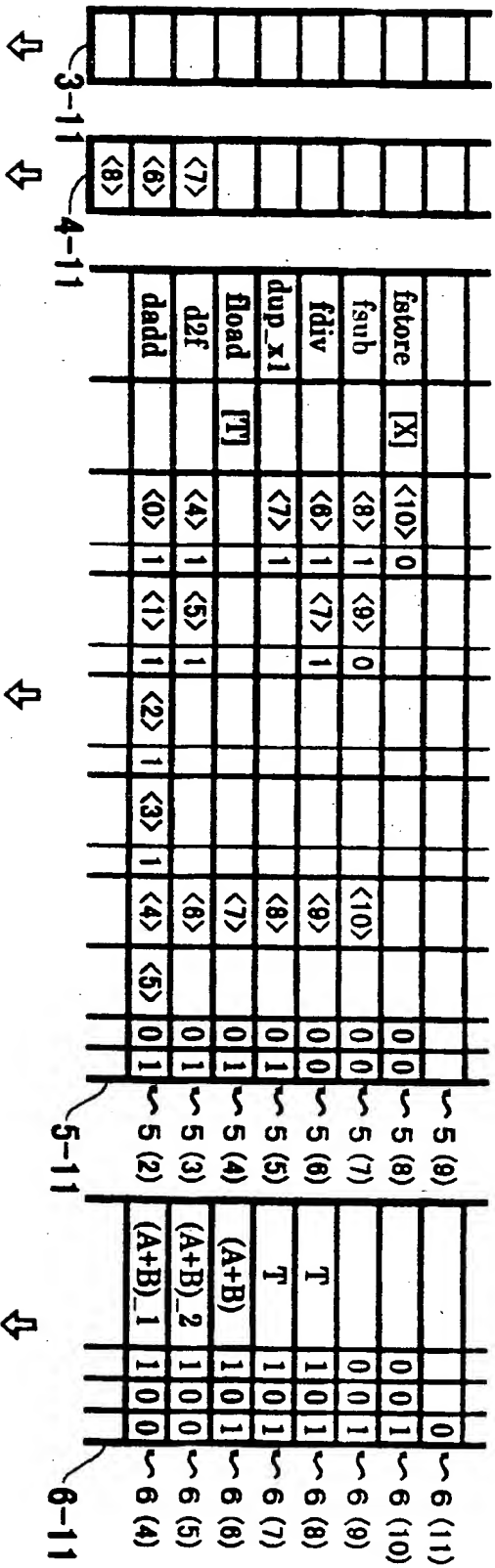
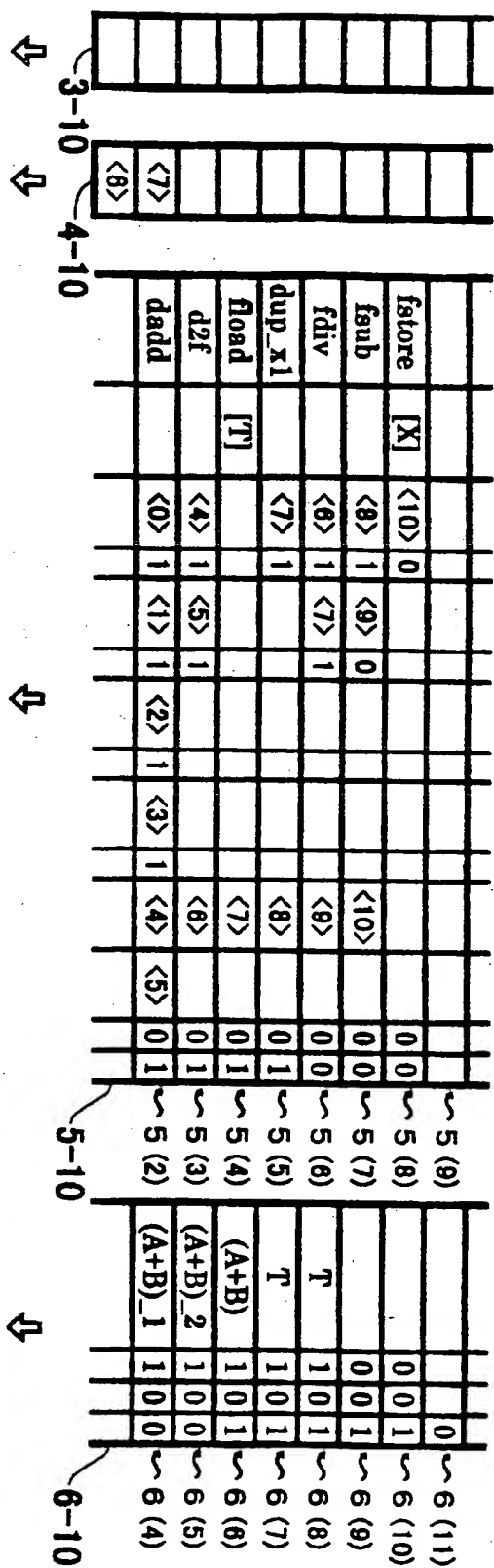
第 8 图



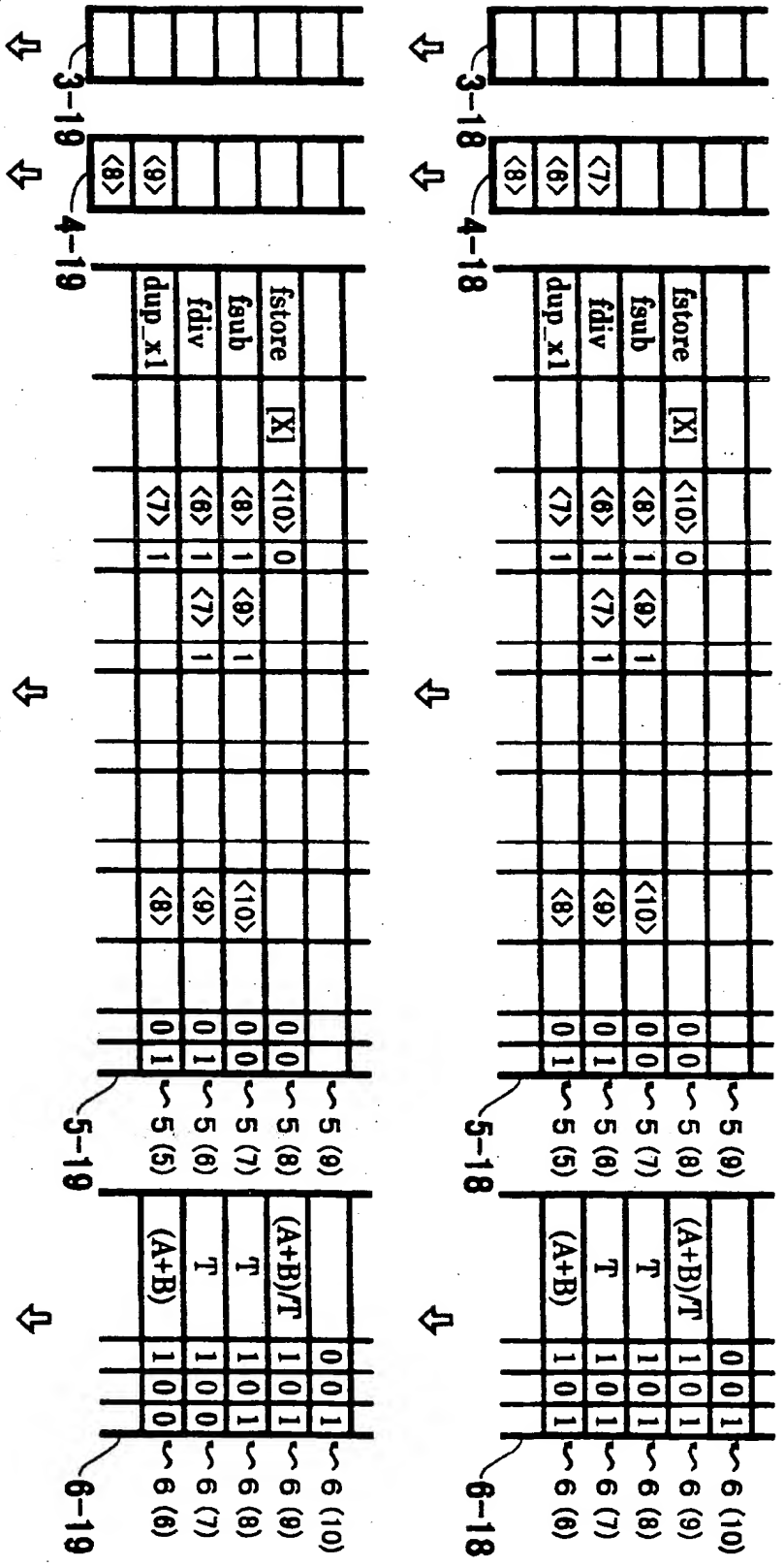
第 9 図



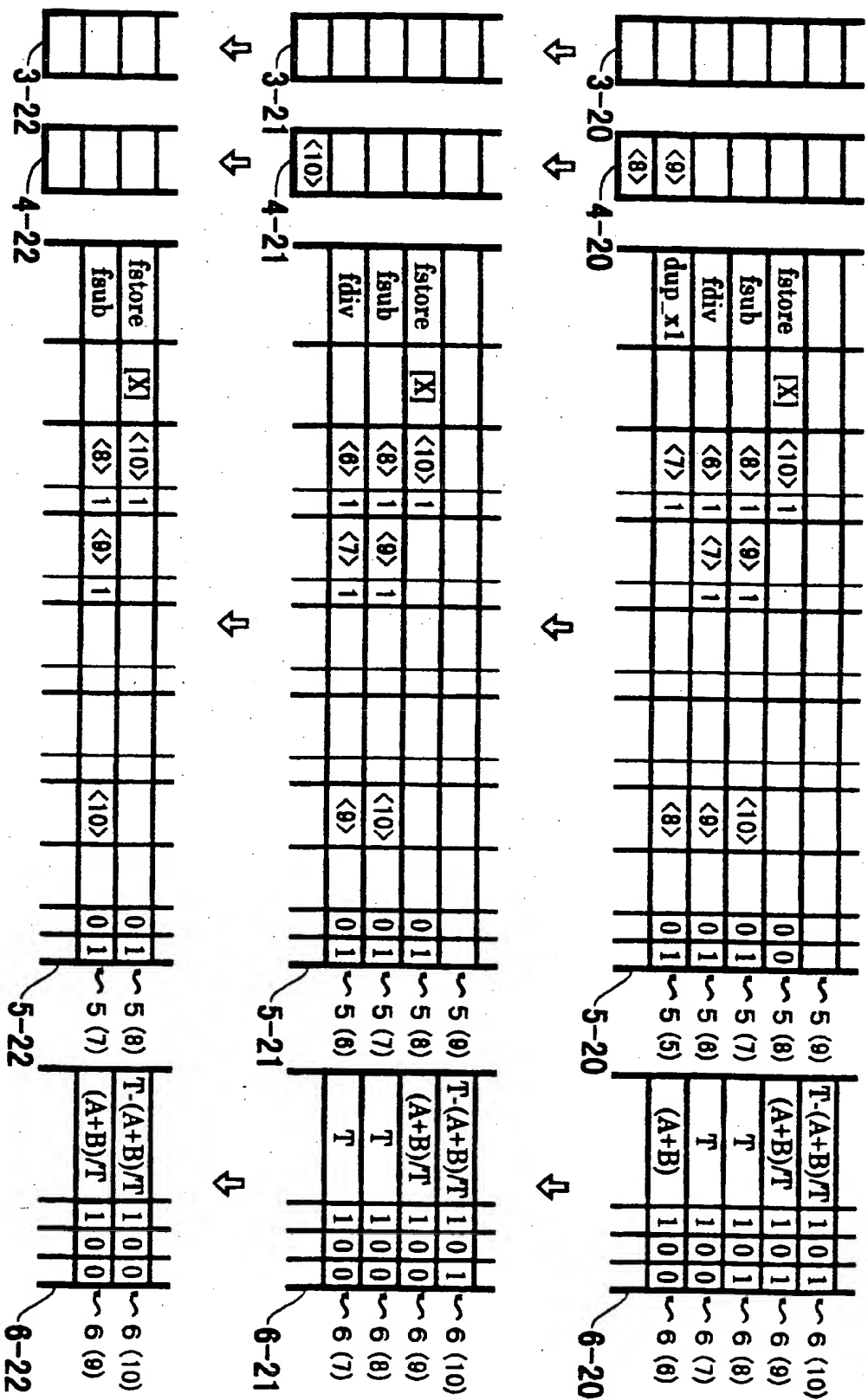
第 10 図



第 11 图



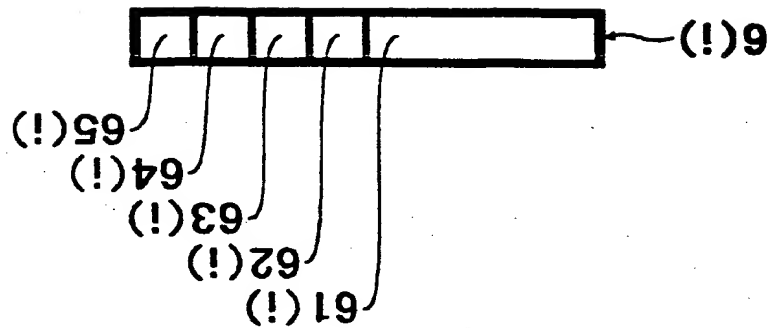
第 12 図



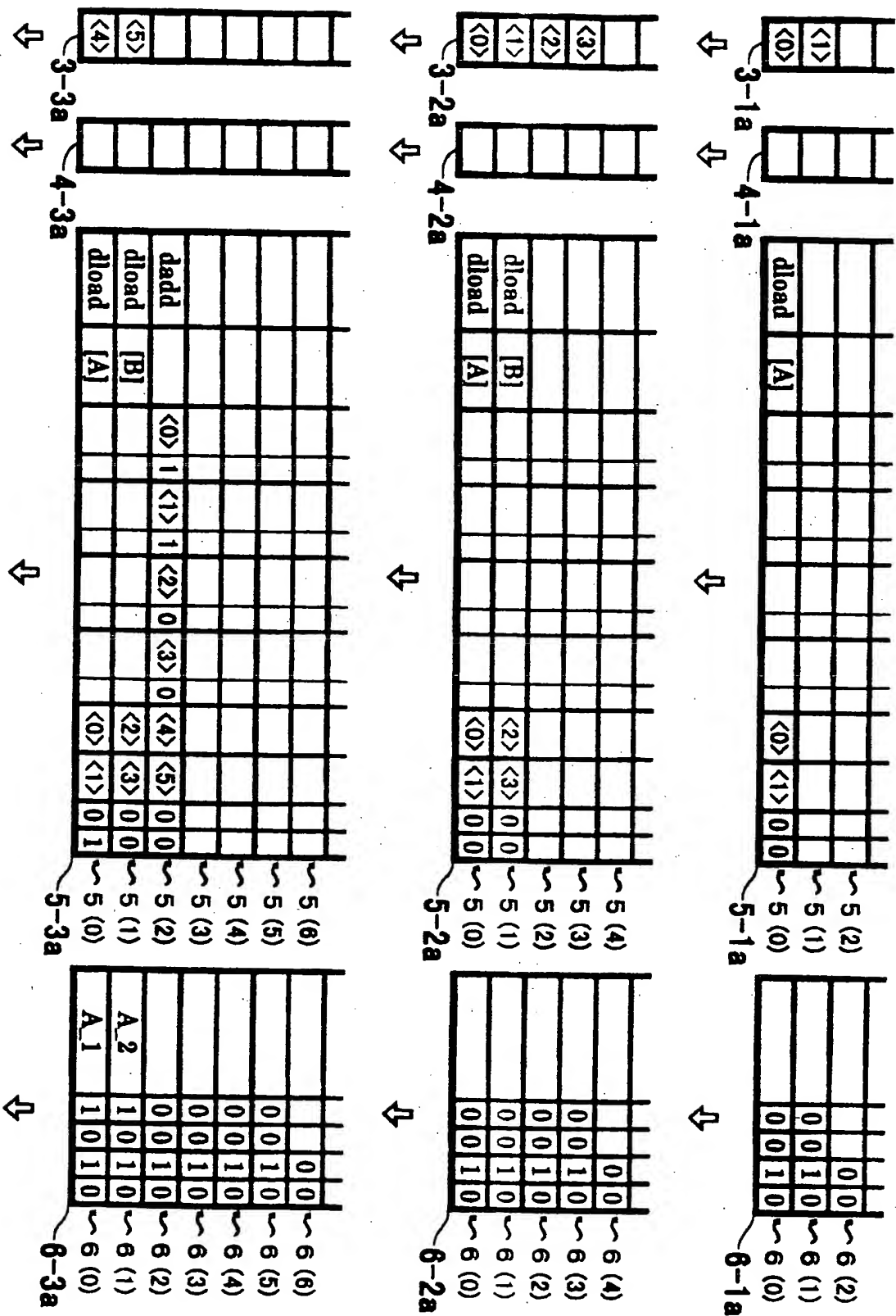
第 1 3 図

PP_OF_APS の増分	A P S の操作内容	I Q の設定内容																																		
+2	<table><tr><td></td><td>NC</td><td>NC</td><td>NC</td><td>f5</td><td>f6</td></tr></table>		NC	NC	NC	f5	f6	<table><tr><td>dload</td><td>[A]</td><td></td><td></td><td></td><td>f1</td><td>f2</td></tr><tr><td>dload</td><td>[B]</td><td></td><td></td><td></td><td>f3</td><td>f4</td></tr><tr><td>dadd</td><td></td><td></td><td>f1</td><td>f2</td><td>f3</td><td>f4</td></tr><tr><td></td><td></td><td></td><td>f5</td><td>f6</td><td></td><td></td></tr></table>	dload	[A]				f1	f2	dload	[B]				f3	f4	dadd			f1	f2	f3	f4				f5	f6		
	NC	NC	NC	f5	f6																															
dload	[A]				f1	f2																														
dload	[B]				f3	f4																														
dadd			f1	f2	f3	f4																														
			f5	f6																																
+1	<table><tr><td></td><td>NC</td><td>NC</td><td>f3</td><td>f1</td><td>f2</td></tr></table>		NC	NC	f3	f1	f2	<table><tr><td>d2f</td><td></td><td>s1</td><td>s0</td><td></td><td>f1</td><td></td></tr><tr><td>load</td><td>[T]</td><td></td><td></td><td></td><td>f2</td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td>f3</td><td></td></tr></table>	d2f		s1	s0		f1		load	[T]				f2							f3								
	NC	NC	f3	f1	f2																															
d2f		s1	s0		f1																															
load	[T]				f2																															
					f3																															
-3	<table><tr><td></td><td>NC</td><td>NC</td><td>NC</td><td>NC</td><td>NC</td></tr></table>		NC	NC	NC	NC	NC	<table><tr><td>fdiv</td><td></td><td>s1</td><td>s0</td><td></td><td>f1</td><td></td></tr><tr><td>fsub</td><td></td><td>s2</td><td>f1</td><td></td><td>f2</td><td></td></tr><tr><td>fstore</td><td>[X]</td><td></td><td>f2</td><td></td><td></td><td></td></tr></table>	fdiv		s1	s0		f1		fsub		s2	f1		f2		fstore	[X]		f2										
	NC	NC	NC	NC	NC																															
fdiv		s1	s0		f1																															
fsub		s2	f1		f2																															
fstore	[X]		f2																																	

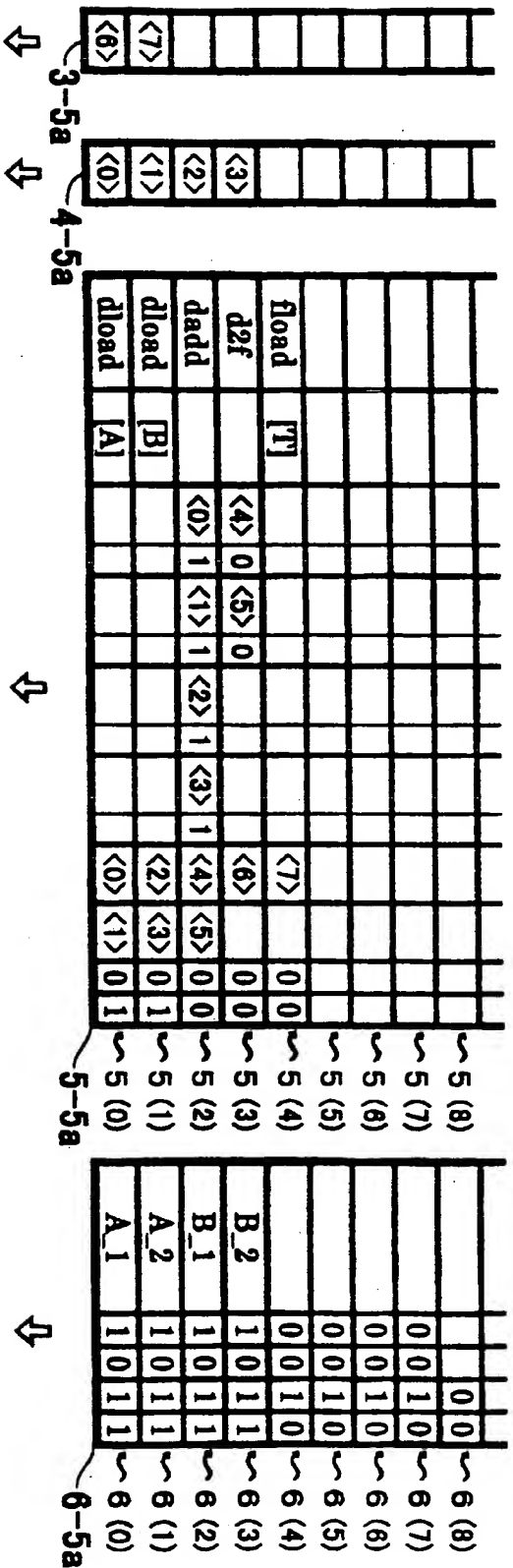
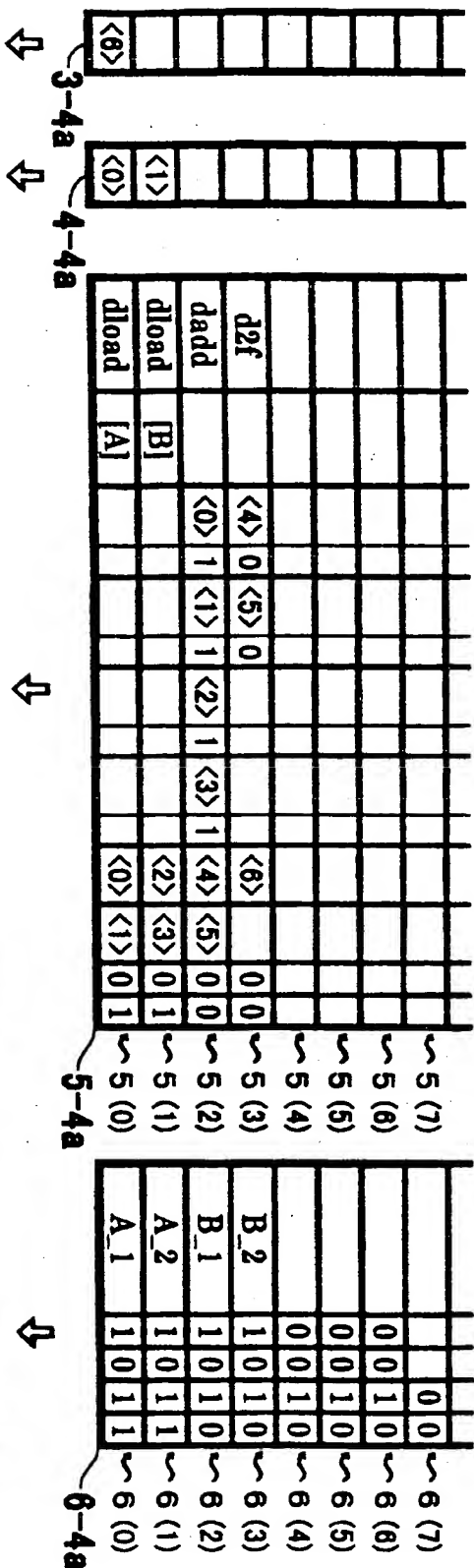
第 1 4 図



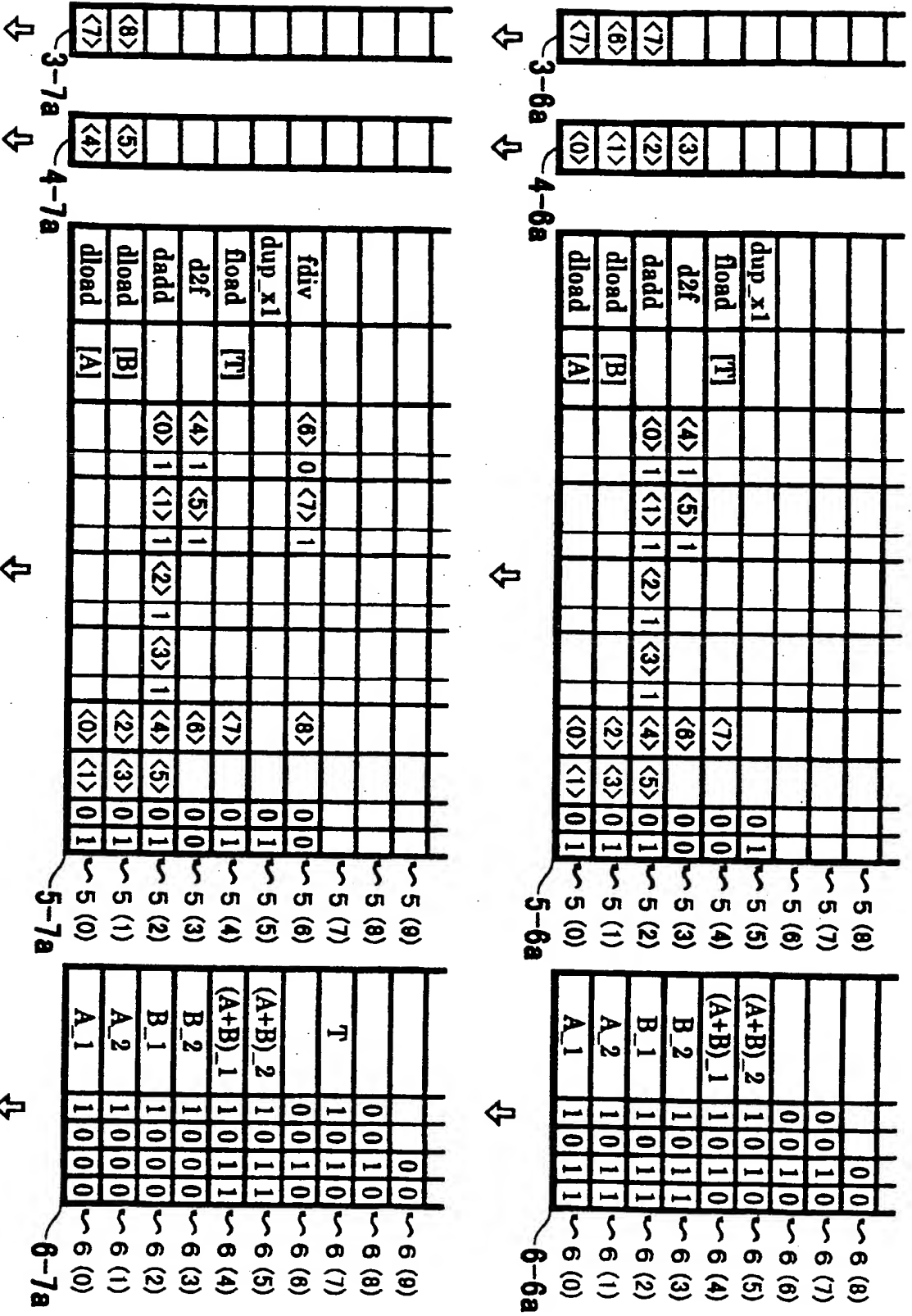
第 15 図



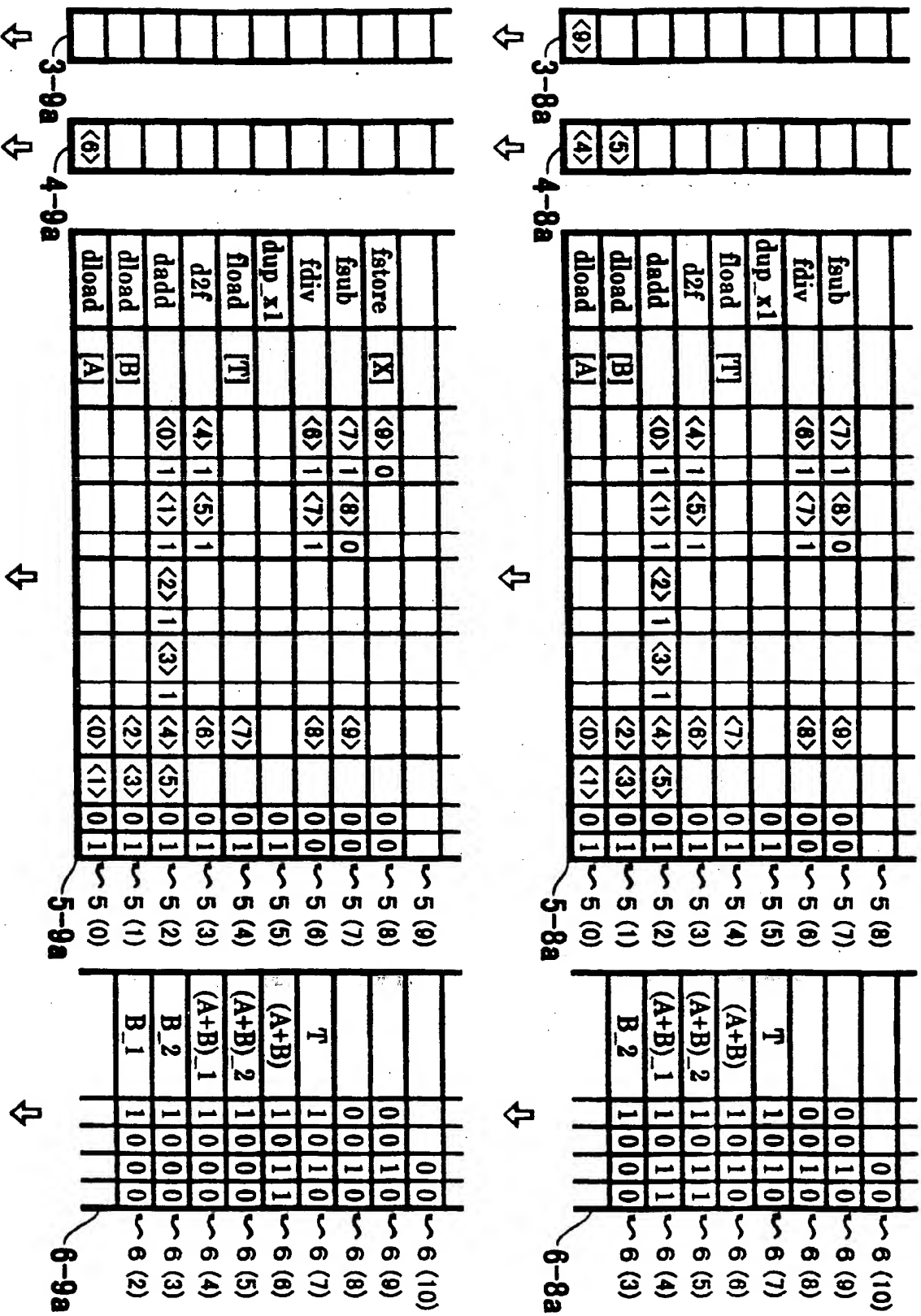
第 16 图



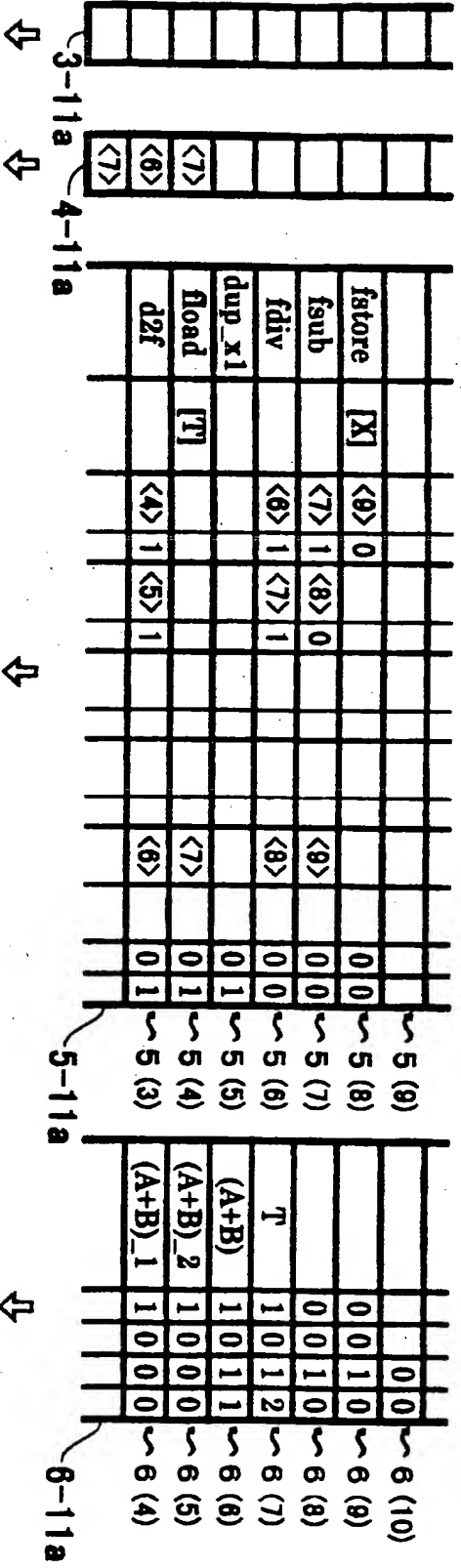
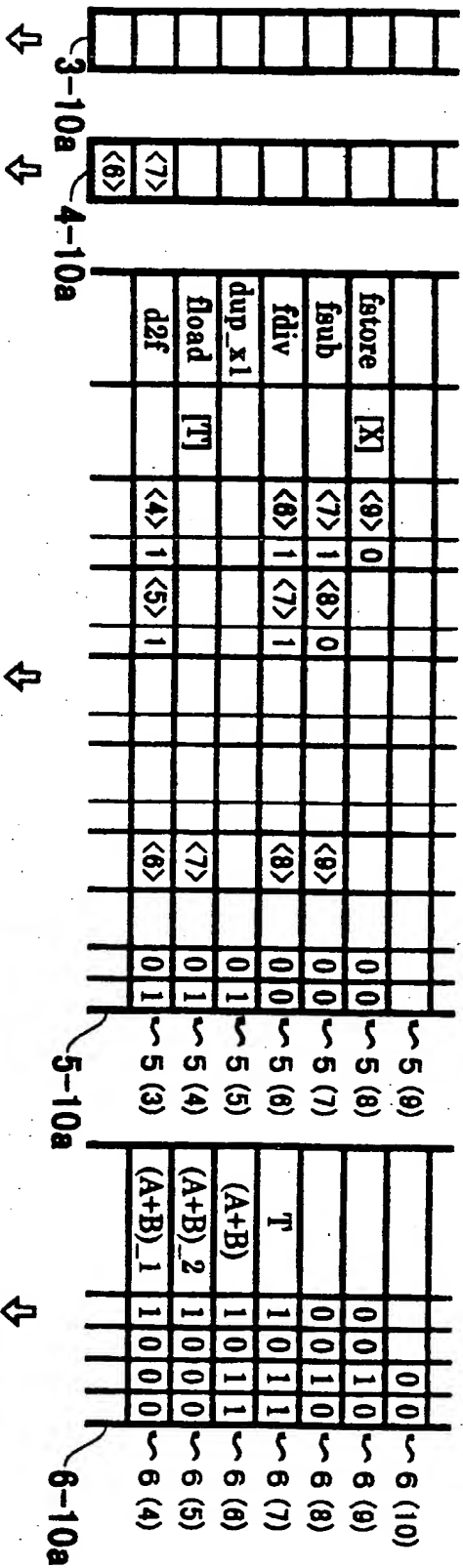
第 1 7 图



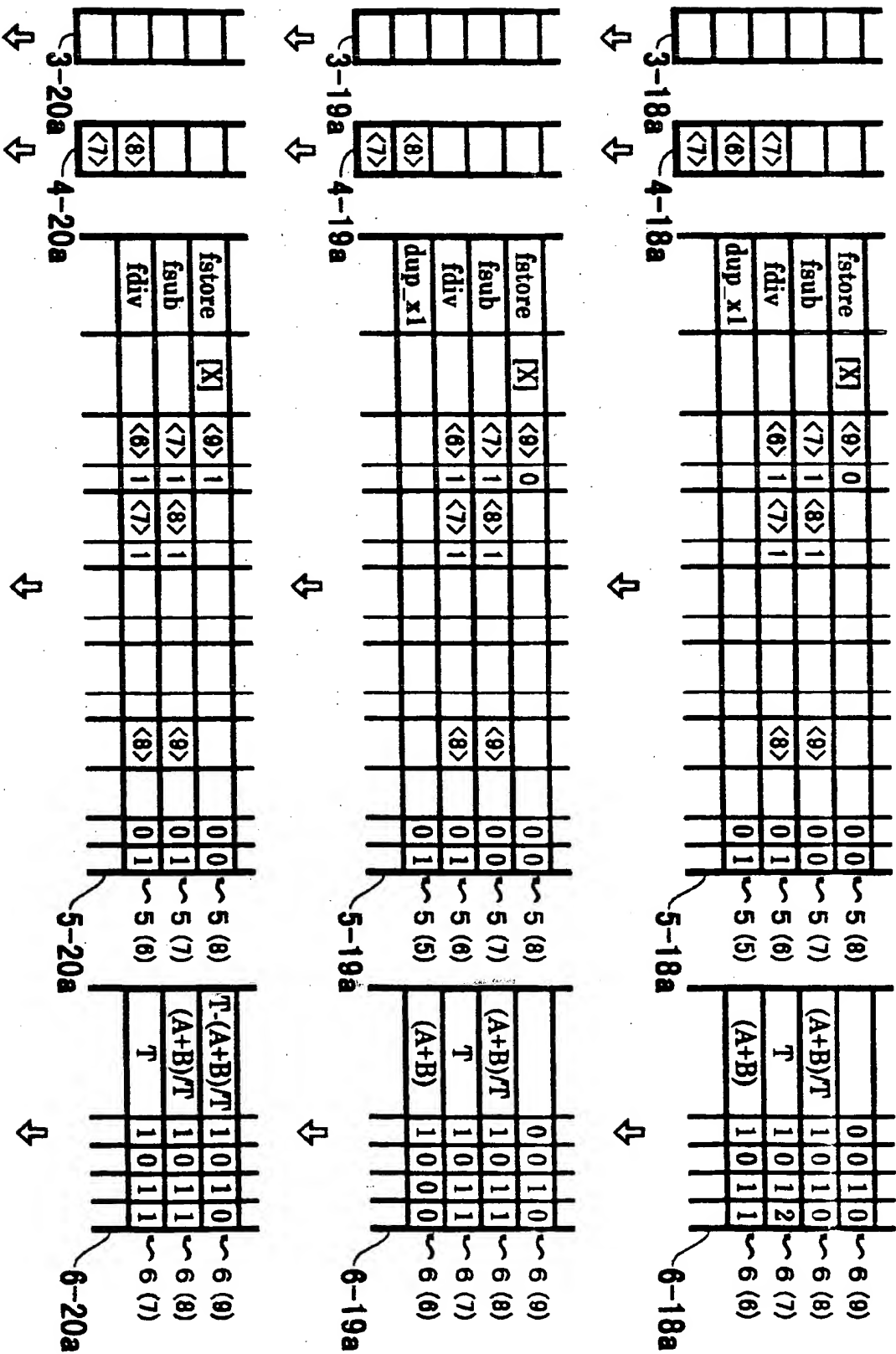
第 18 図



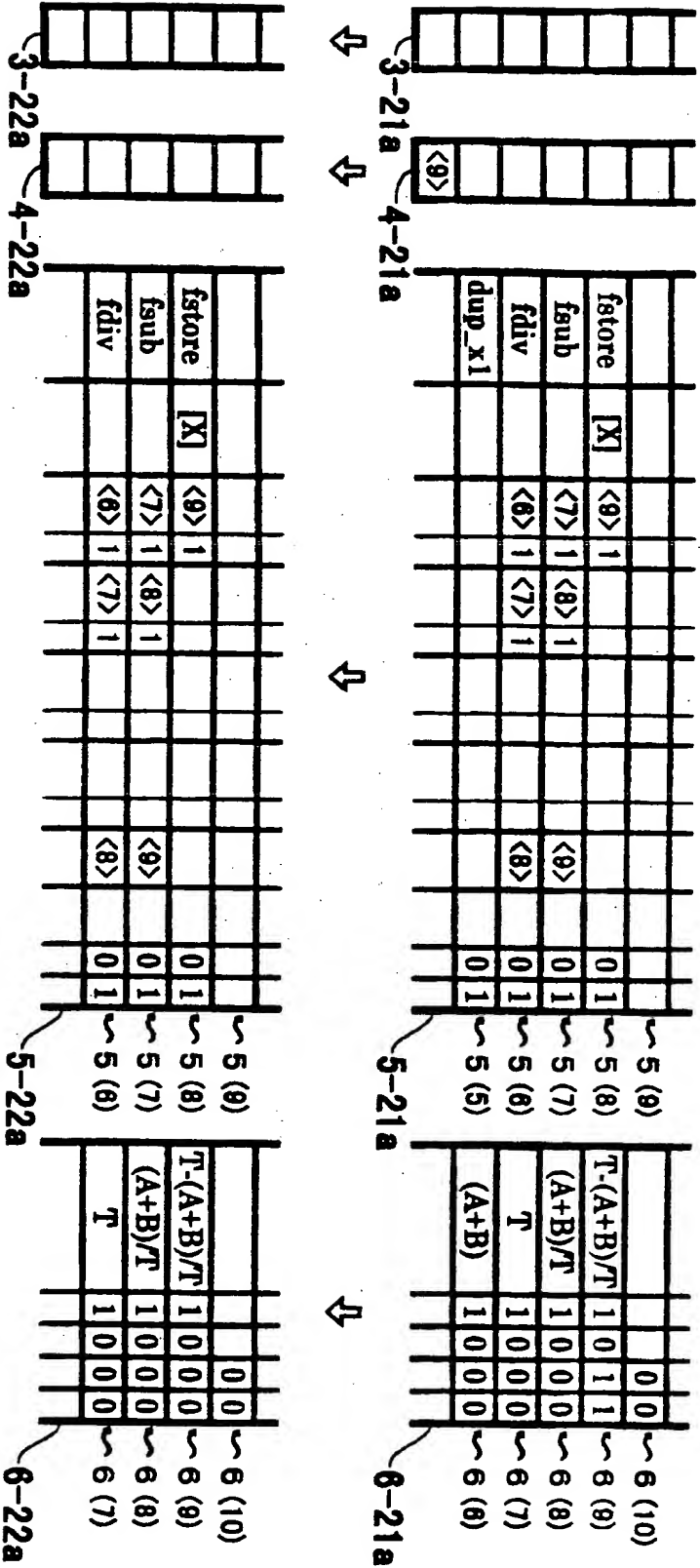
第 19 図



第 20 图



第 2 1 图



Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

PCT/JP98/05230

International application No.

A. CLASSIFICATION OF SUBJECT MATTER Int.Cl. ⁸ G06F9/38, G06F15/82 According to International Patent Classification (IPC) or to both national classification and IPC		B. FIELDS SEARCHED		Minimum documentation searched (classification system followed by classification symbols) Int.Cl. ⁸ G06F9/38, G06F15/82		Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Jitsuyo Shunan Koho 1940-1999 Toroku Jitsuyo Shunan Koho 1994-1999 Kokai Jitsuyo Shunan Koho 1971-1995		Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		C. DOCUMENTS CONSIDERED TO BE RELEVANT	
Category*		Citation of document, with indication, where appropriate, of the relevant passages		Relevant to claim No.		A JP, 2-260082, A (Hajime Seki), 22 October, 1990 (22. 10. 90) (Family: none)		A JP, 62-2330, A (NEC Corp.), 8 January, 1987 (08. 01. 87) (Family: none)		1-6 1-6	
<input type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.											
<p>* Special categories of cited documents:</p> <p>"A" document defining the general state of the art which is not considered to be of particular relevance</p> <p>"E" earlier document but published on or after the international filing date</p> <p>"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)</p> <p>"O" document referring to an oral disclosure, use, exhibition or other means</p> <p>"P" document published prior to the international filing date but later than the priority date claimed</p> <p>"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention</p> <p>"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone</p> <p>"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art</p> <p>"&" document member of the same patent family</p>											
Date of the actual completion of the international search		Date of mailing of the international search report		Name and mailing address of the ISA/ Japanese Patent Office		Facsimile No.		Telephone No.		Authorized officer	
21 January, 1999 (21. 01. 99)		9 February, 1999 (09. 02. 99)									

国際調査報告		国際出願番号 PCT/J P98/05230	
A. 発明の属する分野の分類 (国際特許分類 (IPC)) Int. Cl. G06F 9/38 Int. Cl. G06F15/82			
B. 調査を行った分野 調査を行った最小限資料 (国際特許分類 (IPC)) Int. Cl. G06F 9/38 Int. Cl. G06F15/82 最小限資料以外の資料で調査を行った分野に含まれるもの 日本国実用新案公報 1940-1999年 日本国公開実用新案公報 1971-1995年 日本国登録実用新案公報 1994-1999年			
国際調査で使った電子データベース (データベースの名称、調査に使用した用語) 国際調査すると認められる文献			
引用文献の カテゴリ*		引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	
A	J P, 2-260082, A (関一), 22. 10月. 1990 (22. 10. 90) (フミリーなし)	A	J P, 62-2330, A (日本電気株式会社), 8. 1月. 1987 (08. 01. 87) (フミリーなし)
C. 関連する続きにも文献が列挙されている。 <input type="checkbox"/> パテントファミリーに関する別紙を参照。 <input type="checkbox"/>			
* 引用文献のカテゴリ 「A」 特に関連のある文献ではなく、一般的技術水準を示すもの 「E」 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの 「L」 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す) 「O」 口頭による開示、使用、展示等而言及する文献 「P」 国際出願日前で、かつ優先権の主張の基礎となる出願 「T」 国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの 「X」 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの 「Y」 特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの 「&」 同一パテントファミリー文献			
国際調査を完了した日 21. 01. 99		国際調査報告の発送日 09.02.99	
国際調査機関の名称及びあて先 日本国特許庁 (ISA/J P) 郵便番号100-8915 東京都千代田区霞が関三丁目4番3号		特許庁審査官 (権限のある職員) 中野 裕二 5B 9462	
電話番号 03-3581-1101 内線 3547			

This Page Blank (uspio)